aws

Guide de l'utilisateur

AWS Identity and Access Management

AWS Identity and Access Management: Guide de l'utilisateur

Table of Contents

En quoi consiste IAM ? 1
Vidéo de présentation d'IAM 2
Fonctions d'IAM 2
Accéder à IAM 4
Quand utiliser IAM 5
Lorsque vous effectuez différentes activités professionnelles 5
Lorsque vous êtes autorisé à accéder aux ressources AWS 6
Lorsque vous vous connectez en tant qu'utilisateur IAM 7
Lorsque vous assumez un rôle IAM 7
Lorsque vous créez des politiques et des autorisations 9
Fonctionnement de IAM 10
Conditions 12
Principal 14
Demande 14
Authentification 14
Autorisation 15
Actions ou opérations 16
Ressources 16
Utilisateurs dans AWS 17
Premier accès uniquement : vos informations d'identification utilisateur racine 17
Utilisateurs IAM et utilisateurs dans IAM Identity Center 18
Fédération d'utilisateurs existants 18
Méthodes de contrôle d'accès 20
Autorisations et politiques dans IAM 24
Politiques et comptes 24
Politiques et utilisateurs 25
Politiques et groupes 25
Utilisateurs fédérés et rôles 26
Politiques basées sur les ressources et politiques basées sur l'identité 26
Qu'est-ce que le contrôle d'accès basé sur les attributs (ABAC) ? 27
Comparaison du contrôle ABAC et du modèle RBAC traditionnel 28
Fonctions de sécurité en dehors d'IAM 30
Liens rapides pour les tâches courantes 31
Fonction de recherche de la console IAM 34

Utilisation de la fonction de recherche de la console IAM 35
Icônes dans les résultats de recherche de la console IAM 36
Exemples d'expressions de recherche 36
AWS CloudFormation ressources 37
IAM et modèles AWS CloudFormation 38
En savoir plus sur AWS CloudFormation 38
En utilisant AWS CloudShell 38
Obtention des autorisations IAM pour AWS CloudShell 39
Interaction avec IAM à l'aide de AWS CloudShell 39
Utilisation des AWS SDK 41
Configuration 43
Inscrivez-vous pour un Compte AWS 44
Création d'un utilisateur doté d'un accès administratif 44
Préparation des autorisations de moindre privilège 46
Méthodes de gestion d'IAM 47
AWS Console 47
AWS Interface de ligne de commande (CLI) et kits de développement logiciel (SDK) 49
Votre Compte AWS identifiant et son alias 51
Afficher votre Compte AWS identifiant 51
À propos des alias de compte 53
Création, suppression et affichage d'un alias d' Compte AWS 54
Premiers pas 58
Prérequis 58
Création de votre premier utilisateur IAM 58
Création de votre premier rôle 60
Création de votre première politique IAM 63
Accès par programmation 64
Bonnes pratiques de sécurité et cas d'utilisation 66
Bonnes pratiques de sécurité 66
Obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires 67
Exiger que les charges de travail utilisent des informations d'identification temporaires associées à des rôles IAM pour y accéder AWS 68
Authentification multifactorielle (MFA) nécessaire. 68
Mettre à jour les clés d'accès lorsque cela est nécessaire pour les cas d'utilisation nécessitant des informations d'identification à long terme 69

Suivez les meilleures pratiques pour protéger vos informations d'identification de l'utilisateur root 70
Accorder les autorisations de moindre privilège 70
Commencez avec les politiques AWS gérées et passez aux autorisations du moindre privilège 71
Utiliser IAM Access Analyzer pour générer des politiques de moindre privilège en fonction de l'activité d'accès 71
Vérifiez et supprimez régulièrement les utilisateurs, les rôles, les autorisations, les politiques et les informations d'identification non utilisés 71
Utiliser des conditions dans les stratégies IAM pour restreindre davantage l'accès 72
Vérifiez l'accès public et intercompte aux ressources avec IAM Access Analyzer. 72
Utilisez IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles 72
Établir des garde-fous d'autorisations sur plusieurs comptes 73
Utiliser les limites des autorisations pour déléguer la gestion des autorisations au sein d'un compte. 73
Bonnes pratiques d'utilisateur root 73
Sécurisez vos informations d’identification d'utilisateur root pour empêcher toute utilisation non autorisée 75
Utiliser un mot de passe utilisateur root fort pour protéger l'accès 75
Sécurisez votre connexion d'utilisateur root avec l'authentification multifactorielle (MFA) 76
Ne créer aucune clé d'accès pour l'utilisateur root 76
Utiliser une approbation par plusieurs personnes pour la connexion de l'utilisateur root dans la mesure du possible 77
Utiliser une adresse e-mail de groupe pour les informations d'identification de l'utilisateur root 77
Restreindre l'accès aux mécanismes de récupération des comptes 77
Sécurisez les informations d’identification d'utilisateur root de votre compte Organizations 78
Surveiller l'accès et l'utilisation 79
Cas d'utilisation métier 80
Configuration initiale d'Example Corp 81
Cas d'utilisation pour IAM avec Amazon EC2 82
Cas d'utilisation pour IAM avec Amazon S3 83
Didacticiels 86
Accorder l'accès à la console de facturation 86
Prérequis 88

Étape 1 : Activez l'accès IAM aux informations de facturation sur votre compte de test AWS 88
Étape 2 : Créer des utilisateurs et des groupes de test 89
Étape 3 : Créer un rôle pour accorder l'accès à la console AWS Billing 91
Étape 4 : Tester l'accès à la console 93
Récapitulatif 94
Ressources connexes 94
Déléguer l'accès Comptes AWS aux différents rôles 95
Prérequis 97
Créer un rôle dans le compte production 98
Accorder l'accès au rôle 101
Tester l'accès en changeant de rôles 103
Ressources connexes 109
Récapitulatif 110
Créer une politique gérée par le client 110
Prérequis 111
Étape 1 : Créer la politique 111
Étape 2 : Attacher la politique 112
Étape 3 : Tester l'accès utilisateur 113
Ressources connexes 113
Récapitulatif 113
Utiliser le contrôle d'accès basé sur les attributs (ABAC, attribute-based access control) 114
Présentation du didacticiel 115
Prérequis 116
Étape 1 : Créer des utilisateurs test 117
Étape 2 : Créer la politique ABAC 119
Étape 3 : Créer les rôles 123
Étape 4 : Tester la création de secrets 125
Étape 5 : Tester l'affichage des secrets 128
Étape 6 : Tester l'adaptabilité 130
Étape 7 : Tester la mise à jour et la suppression des secrets 132
Récapitulatif 134
Ressources connexes 134
Utilisation de balises de session SAML pour le contrôle ABAC 135
Autoriser les utilisateurs à gérer leurs informations d'identification et leurs paramètres MFA 140
Prérequis 141

Étape 1 : Créer une politique pour appliquer l'authentification MFA 142
Étape 2 : Attacher des politiques à votre groupe d'utilisateurs test 143
Étape 3 : Test de l'accès utilisateur 144
Ressources connexes 146
Identités 147
Compte AWS utilisateur root 148
Utilisateurs IAM 148
Groupes d'utilisateurs IAM 149
Rôles IAM 149
Informations d'identification temporaires dans IAM 151
Quand utiliser les utilisateurs IAM Identity Center ? 151
Quand créer un utilisateur IAM (au lieu d'un rôle) 152
Quand créer un rôle IAM (au lieu d'un utilisateur) 153
Comparer Utilisateur racine d'un compte AWS et utiliser IAM 154
Utilisateur racine d'un compte AWS 155
Activez le MFA pour votre Utilisateur racine d'un compte AWS (console) 156
Modifier le mot de passe 165
Réinitialisation d'un mot de passe d'utilisateur racine perdu ou oublié 167
Création de clés d'accès pour l'utilisateur racine 168
Suppression des clés d'accès pour l'utilisateur racine 171
Tâches nécessitant l'utilisateur root 173
Résolution des problèmes d'utilisateur root 175
Informations connexes 176
Users 176
Comment AWS identifier un utilisateur IAM 177
Utilisateurs IAM et informations d'identification 177
Rôles et autorisations IAM 179
Utilisateurs et comptes IAM 179
Utilisateurs IAM en tant que comptes de service 180
Ajout d'un utilisateur 180
Contrôle de l'accès utilisateur à la console 188
Comment les utilisateurs d'IAM se connectent à AWS 190
Gestion des utilisateurs 194
Modification des autorisations pour un utilisateur 201
Gestion des mots de passe 209
Clés d'accès 229

Récupération des mots de passe ou clés d'accès perdus ... 247
Authentification multifactorielle (MFA) ... 249
Trouver les informations d'identification non utilisées ... 329
Obtention de rapports d'informations d'identification ... 333
Utilisation d'IAM avec CodeCommit ... 340
Utilisation d'IAM avec Amazon Keyspaces ... 344
Gestion des certificats de serveur ... 346
Groupes d’utilisateurs ... 353
Création de groupes d'utilisateurs ... 355
Gestion des groupes d'utilisateurs ... 357
Rôles ... 365
Termes et concepts ... 366
Scénarios courants ... 371
Rôles liés à un service ... 391
Créer des rôles ... 405
Utilisation de rôles ... 446
Gestion des rôles ... 623
Fournisseurs d'identité et fédération ... 648
Fédération avec IAM Identiy Center ... 650
Fédération avec IAM ... 650
Fédération avec les réserves d'identités Amazon Cognito ... 651
Scénarios courants ... 652
Fédération OIDC ... 658
Fédération SAML 2.0 ... 678
informations d’identification de sécurité temporaires ... 711
AWS STS et AWS régions ... 712
Scénarios courants d'informations d'identification temporaires ... 713
Demande d'informations d'identification temporaires de sécurité ... 715
Utilisation d'informations d'identification temporaires avec des ressources AWS ... 734
Contrôle des autorisations affectées aux informations d'identification de sécurité temporaires ... 740
Gérer AWS STS dans un Région AWS ... 775
Utilisation des jetons porteurs ... 786
Exemples d'application qui utilisent des informations d'identification temporaires ... 787
Activation de l'accès à la AWS console par un courtier d'identité personnalisé ... 788
Ressources supplémentaires pour les informations d'identification temporaires ... 803

Balisage des ressources IAM 804
Choisissez une convention de dénomination des AWS balises 805
Règles de balisage dans IAM et AWS STS 806
Étiquette d'utilisateurs IAM 810
Étiquette de rôles IAM 813
Balisage des politiques gérées par le client 816
Balisage de fournisseurs d'identité IAM 819
Balisage de profils d'instance 826
Balisage des certificats de serveur 828
Balisage des dispositifs MFA virtuels 831
Balises de session 834
Enregistrez les événements avec CloudTrail 849
IAM et AWS STS informations dans CloudTrail 849
Journalisation des demandes IAM et AWS STS API 850
Journalisation des demandes d'API vers d'autres services AWS 851
Journalisation des événements de connexion utilisateur 851
Journalisation des événements de connexion pour les informations d'identification temporaires 852
Exemple d'événements d'API IAM dans le journal CloudTrail 855
Exemples AWS STS d'événements d'API dans le CloudTrail journal 856
Exemple d'événements de connexion dans le journal CloudTrail 865
Comportement de la politique de confiance dans les rôles IAM 868
Gestion des accès 870
Ressources de gestion des accès 871
Politiques et autorisations 872
Types de politique 872
Politiques et utilisateur racine 878
Présentation des politiques JSON 878
Accorder les privilèges les plus faibles possible 884
Politiques gérées et politiques en ligne 886
Périmètres de données 897
Limites d'autorisations 902
Identité et ressource 916
Contrôle de l'accès à l'aide de politiques 920
Contrôlez l'accès aux utilisateurs et rôles IAM à l'aide de balises 933
Contrôlez l'accès aux AWS ressources à l'aide de balises 936

Accès intercompte aux ressources 941
Transmission des sessions d'accès 948
Exemples de politiques 951
Gestion des politiques IAM 1033
Création de politiques IAM 1034
Validation des politiques 1045
Générer des politiques 1046
Test des politiques IAM 1047
Ajouter ou supprimer des autorisations d'identité 1064
Gestion des versions des politiques IAM 1077
Modification de politiques IAM 1082
Suppression de politiques IAM 1088
Ajustement des autorisations à l'aide des informations consultées 1093
Comprendre les politiques 1640
Récapitulatif de la politique (liste des services) 1641
Récapitulatif du service (liste des actions) 1655
Récapitulatif de l'action (liste des ressources) 1661
Exemples de récapitulatifs de la politique 1665
Autorisations nécessaires 1675
Autorisations pour la gestion des identités IAM 1675
Autorisations pour utiliser AWS Management Console 1677
Octroi d'autorisations dans plusieurs comptes AWS 1678
Autorisations pour qu'un service accède à un autre service 1678
Actions requises 1679
Exemples de politiques pour IAM 1680
Exemples de code 1685
IAM 1690
Actions 1705
Scénarios 2276
AWS STS 2631
Actions 2632
Scénarios 2661
Sécurité 2680
AWS informations d'identification de sécurité 2681
Considérations sur la sécurité 2682
Identité fédérée 2683

Authentification multifactorielle (MFA) 2683
Accès par programmation 2684
Alternatives aux clés d'accès à long terme 2686
Accès à AWS l'aide de vos AWS informations d'identification 2688
AWS directives relatives aux audits de sécurité 2688
Quand effectuer un audit de sécurité ? 2689
Consignes pour l'audit 2689
Vérifiez les informations d'identification AWS de votre compte 2690
Examen de vos utilisateurs IAM 2690
Examen de vos groupes IAM 2691
Examen de vos rôles IAM 2691
Examen de vos fournisseurs IAM pour SAML et OpenID Connect (OIDC) 2691
Examen de vos applications mobiles 2692
Conseils pour l'examen des politiques IAM 2692
Protection des données 2694
Chiffrement des données dans IAM et AWS STS 2695
Gestion des clés dans IAM et AWS STS 2696
Confidentialité du trafic interréseau dans IAM et AWS STS 2696
Journalisation et surveillance 2696
Validation de conformité 2697
Résilience 2699
Bonnes pratiques pour la résilience IAM 2701
Sécurité de l’infrastructure 2701
Analyse de la configuration et des vulnérabilités 2702
AWS politiques gérées 2703
Accès IAM ReadOnly 2704
Mot de passe IAM UserChange 2704
IAM AccessAnalyzer FullAccess 2705
Accès IAM AccessAnalyzer ReadOnly 2706
AccessAnalyzerServiceRolePolitique 2707
..... 2710
Mises à jour des politiques 2710
IAM Access Analyzer 2715
Identification des ressources partagées avec une entité externe 2715
Identification des accès non utilisés accordés aux utilisateurs et aux rôles IAM 2718
Validation des politiques par rapport aux bonnes pratiques AWS 2718

Validation des politiques par rapport aux normes de sécurité que vous avez spécifiées 2719
Générer des politiques 2719
Tarification pour l’analyseur d’accès IAM 2719
Résultats des accès externes et non utilisés 2720
Comment fonctionnent les résultats de l’IAM Access Analyzer 2722
Démarrage avec l’IAM Access Analyzer 2723
Tableau de bord de résultats 2730
Utilisation des résultats 2734
Examen des résultats 2735
Filtrage des résultats 2739
Archivage des résultats 2744
Résolution des résultats 2745
Types de ressources pris en charge 2748
Paramètres 2756
Règles d'archivage 2759
Surveillance avec EventBridge 2761
Intégration avec Security Hub 2771
Se connecter avec CloudTrail 2778
Clés de filtre de l’IAM Access Analyzer 2782
Utilisation des rôles liés aux services 2791
Prévisualiser l'accès 2794
Prévisualiser l'accès dans la console Amazon S3 2795
Prévisualiser l'accès avec les API de l’IAM Access Analyzer 2796
Vérifications de validation des politiques 2800
Validation de la politique de l’IAM Access Analyzer 2801
Vérifications de politique personnalisées 2909
Génération d'une politique IAM Access Analyzer 2914
Processus de génération de politique 2914
Informations de niveau service et action 2915
À savoir 2915
Autorisations nécessaires 2917
Génération d'une politique basée sur CloudTrail l'activité (console) 2920
Générer une politique en utilisant AWS CloudTrail les données d'un autre compte 2924
Génération d'une politique basée sur CloudTrail l'activité (AWS CLI) 2927
Générer une politique basée sur CloudTrail l'activité (AWS API) 2928
Services de génération d'une politique d'Analyseur d'accès IAM 2928

Quotas de l'IAM Access Analyzer 2939
Dépannage IAM 2942
Problèmes généraux 2942
Je ne peux pas me connecter à mon compte AWS 2943
J'ai perdu mes clés d'accès 2943
Les variables de la politique ne fonctionnent pas 2943
Les modifications que j'apporte ne sont pas toujours visibles immédiatement 2944
Je ne suis pas autorisé à effectuer : iam : MFAdevice DeleteVirtual 2945
Comment créer des utilisateurs IAM en toute sécurité ? 2945
Ressources supplémentaires 2946
Messages d'erreur d'accès refusé 2947
Je reçois un « accès refusé » lorsque je fais une demande à un AWS service 2947
Je reçois un message d'accès refusé lorsque j'effectue une demande avec des informations d'identification de sécurité temporaires 2949
Exemples d'accès refusé 2951
Politiques IAM 2956
Résolution des problèmes à l'aide de l'éditeur visuel 2958
Résolution des problèmes à l'aide des récapitulatifs de politique 2963
Résolution des problèmes de gestion des politiques 2973
Résolution des problèmes de documents de politique JSON 2974
Clés de sécurité FIDO 2980
Je ne peux pas activer ma clé de sécurité FIDO 2980
Je ne peux pas me connecter à l'aide de ma clé de sécurité FIDO 2982
J'ai perdu ou cassé ma clé de sécurité FIDO 2982
Autres problèmes 2982
Rôles IAM 2982
Je ne parviens pas à endosser un rôle 2983
Un nouveau rôle est apparu dans mon compte AWS 2985
Je ne parviens pas à modifier ou supprimer un rôle dans mon Compte AWS 2986
Je ne suis pas autorisé à exécuter : iam : PassRole 2986
Pourquoi ne puis-je pas endosser un rôle avec une session de 12 heures ? (AWS CLI, AWS API) 2987
Je reçois une erreur lorsque j'essaie de changer de rôle dans la console IAM 2987
Mon rôle dispose d'une politique qui me permet d'effectuer une action, mais j'obtiens « Accès refusé » 2988
Le service n'a pas créé la version de politique par défaut du rôle 2988

Il n'existe aucun cas d'utilisation pour un rôle de service dans la console 2990
IAM et Amazon EC2 2991
Lors de la tentative de lancement de l'instance, je ne vois pas le rôle attendu dans la liste de Rôles IAM de la console Amazon EC2. 2991
Les informations d'identification sur mon instance concernent le mauvais rôle 2992
Quand je tente d'appeler `AddRoleToInstanceProfile`, je reçois un message d'erreur `AccessDenied`. 2993
Amazon EC2 : quand je tente de lancer l'instance avec un rôle, je reçois un message d'erreur `AccessDenied` 2993
Je ne parviens pas à accéder aux informations d'identification de sécurité temporaires sur mon instance EC2 2994
Que signifient les erreurs dans le document `info` de la sous-arborescence IAM ? 2995
IAM et Amazon S3 2996
Comment accorder un accès anonyme à un compartiment Amazon S3 ? 2996
Je suis connecté en tant qu'utilisateur Compte AWS root ; pourquoi ne puis-je pas accéder à un compartiment Amazon S3 sous mon compte ? 2996
Fédération SAML 2.0 2997
Réponse SAML non valide 2998
RoleSessionName est obligatoire 2998
Non autorisé pour le AssumeRoleWith SAML 2999
RoleSessionName Caractères non valides 3000
Caractères d'identité source non valides 3000
Signature de réponse non valide 3000
Impossibilité d'endosser un rôle 3001
Impossible d'analyser les métadonnées 3001
Le fournisseur spécifié n'existe pas 3001
DurationSeconds dépasse MaxSessionDuration 3002
La réponse ne contient pas l'audience requise 3002
Affichage d'une réponse SAML dans votre navigateur 3002
Référence 3007
Amazon Resource Names (ARN) 3007
Format ARN 3007
Rechercher le format d'ARN pour une ressource 3009
Chemins d'accès dans les ARN 3009
Identifiants IAM 3010
Noms conviviaux et chemins 3010

ARN IAM 3011
Identifiants uniques 3018
IAM et quotas AWS STS 3021
Exigences relatives aux noms IAM 3021
Quotas d'objet IAM 3022
Quotas de l'IAM Access Analyzer 3024
Quotas des rôles Anywhere IAM 3024
Limites des caractères d'IAM et de STS 3024
Points de terminaison de VPC d’Interface 3030
Disponibilité 3031
Créez un point de terminaison VPC pour AWS STS 3032
Services qui fonctionnent avec IAM 3033
Services qui fonctionnent avec IAM 3035
En savoir plus 3108
Signature des demandes AWS d'API 3112
Quand signer des demandes ? 3114
Pourquoi les demandes sont-elles signées ? 3114
Éléments d'une demande Signature Version 4 3115
Méthodes d'authentification 3117
Création d’une requête signée 3122
Demander des exemples de signature 3134
Dépannage 3136
Référence de politique 3140
Références des éléments JSON 3141
Logique d'évaluation de politiques 3216
Syntaxe de politique 3240
AWS politiques gérées pour les fonctions professionnelles 3249
Clés de condition globale 3267
Clés de condition IAM 3331
Actions, ressources et clés de condition 3362
Ressources 3363
Identités 3363
Informations d'identification (mots de passe, clés d'accès et dispositifs MFA) 3363
Autorisations et politiques 3364
Fédération et délégation 3365
IAM et autres produits AWS 3365

Utilisation d'IAM avec Amazon EC2 3365
Utilisation d'IAM avec Amazon S3 3365
Utilisation d'IAM avec Amazon RDS 3366
Utilisation d'IAM avec Amazon DynamoDB 3366
Pratiques de sécurité générales 3366
Ressources générales 3367
Envoi de demandes de requête HTTP 3369
Points de terminaison 3370
HTTPS requis 3370
Signature des demandes d'API IAM 3370
Historique de la documentation 3372
..... mmmcccxcvii

En quoi consiste IAM ?

Follow us on Twitter

AWS Identity and Access Management (IAM) est un service Web qui vous permet de contrôler en toute sécurité l'accès aux AWS ressources. Avec IAM, vous pouvez gérer de manière centralisée les autorisations qui contrôlent les AWS ressources auxquelles les utilisateurs peuvent accéder. Vous pouvez utiliser IAM pour contrôler les personnes qui s'authentifient (sont connectées) et sont autorisées (disposent d'autorisations) à utiliser des ressources.

Lorsque vous créez un Compte AWS, vous commencez par une identité de connexion unique qui donne un accès complet à toutes Services AWS les ressources du compte. Cette identité est appelée utilisateur Compte AWS root et est accessible en vous connectant avec l'adresse e-mail et le mot de passe que vous avez utilisés pour créer le compte. Il est vivement recommandé de ne pas utiliser l'utilisateur racine pour vos tâches quotidiennes. Protégez vos informations d'identification d'utilisateur racine et utilisez-les pour effectuer les tâches que seul l'utilisateur racine peut effectuer. Pour obtenir la liste complète des tâches qui nécessitent que vous vous connectiez en tant qu'utilisateur root, veuillez consulter Tâches nécessitant les informations d'identification de l'utilisateur root.

Table des matières

- Vidéo de présentation d'IAM
- Fonctions d'IAM
- Accéder à IAM
- Quand utiliser IAM ?
- Fonctionnement de IAM
- Vue d'ensemble de la gestion des AWS identités : utilisateurs
- Présentation de la gestion des accès : autorisations et politiques
- À quoi sert ABAC ? AWS
- Fonctions de sécurité en dehors d'IAM
- Liens rapides pour les tâches courantes
- Fonction de recherche de la console IAM
- Création de AWS Identity and Access Management ressources avec AWS CloudFormation

- Utilisation AWS CloudShell pour travailler avec AWS Identity and Access Management
- Utilisation d'IAM avec un SDK AWS

Vidéo de présentation d'IAM

AWS Training and Certification propose une présentation vidéo de 10 minutes de l'IAM :

Introduction à AWS Identity and Access Management

Fonctions d'IAM

IAM vous offre les fonctions suivantes :

Accès partagé à votre Compte AWS

Vous pouvez accorder à d'autres utilisateurs l'autorisation d'administrer et d'utiliser les ressources de votre compte AWS sans avoir à partager votre mot de passe ou clé d'accès.

Autorisations granulaires

Vous pouvez accorder différentes autorisations à différents utilisateurs concernant différentes ressources. Par exemple, vous pouvez autoriser certains utilisateurs à accéder totalement à Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB, Amazon Redshift et à d'autres services. AWS Pour d'autres utilisateurs, vous pouvez autoriser un accès en lecture seule à certains compartiments S3 ou l'autorisation d'administrer certaines instances EC2 seulement, ou encore un accès à vos informations de facturation uniquement.

Accès sécurisé aux AWS ressources pour les applications exécutées sur Amazon EC2

Vous pouvez utiliser les fonctions IAM pour fournir en toute sécurité des informations d'identification pour les applications s'exécutant sur des instances EC2. Ces informations d'identification permettent à votre application d'accéder à d'autres AWS ressources. Il s'agit notamment des compartiments S3 et des tables DynamoDB.

Authentification multifactorielle (MFA)

Vous pouvez ajouter une authentification à deux facteurs à votre compte et aux utilisateurs individuels pour une sécurité renforcée. Avec l'authentification MFA, vos utilisateurs ou vous-

même fournissez non seulement un mot de passe ou une clé d'accès pour utiliser votre compte, mais aussi un code généré par un dispositif configuré spécialement. Si vous utilisez déjà une clé de sécurité FIDO avec d'autres services et que sa configuration est AWS prise en charge, vous pouvez l'utiliser WebAuthn pour la sécurité MFA. Pour plus d'informations, consultez Configurations prises en charge pour l'utilisation des clés d'accès et des clés de sécurité.

Fédération des identités

Vous pouvez autoriser des utilisateurs disposant déjà de mots de passe ailleurs, par exemple, dans votre réseau d'entreprise ou auprès d'un fournisseur d'identité Internet, à obtenir un accès temporaire à votre Compte AWS.

Informations d'identité par sécurité

Si vous utilisez AWS CloudTrail, vous recevez des enregistrements de journaux incluant des informations sur les utilisateurs effectuant des demandes concernant les ressources de votre compte. Ces informations sont basées sur les identités IAM.

Conformité PCI DSS

IAM prend en charge le traitement, le stockage et la transmission des données de cartes bancaires par un commerçant ou un fournisseur de services et a été validé comme étant conforme à la norme PCI (Payment Card Industry) DSS (Data Security Standard). Pour plus d'informations sur la norme PCI DSS, notamment sur la manière de demander une copie du Package de AWS conformité PCI, consultez la section PCI DSS niveau 1.

Intégré à de nombreux AWS services

Pour obtenir la liste des AWS services compatibles avec IAM, consultezAWS services qui fonctionnent avec IAM.

Cohérence à terme

L'IAM, comme de nombreux autres AWS services, est finalement cohérent. IAM garantit une haute disponibilité en répliquant les données sur plusieurs serveurs dans les centres de données d'Amazon du monde entier. Si une demande de modification de certaines données aboutit, la modification est validée et stockée en toute sécurité. En revanche, cette modification doit être répliquée dans IAM, ce qui peut prendre un certain temps. Les modifications peuvent être la création ou la mise à jour d'utilisateurs, de groupes, de rôles ou de politiques. Nous vous recommandons de ne pas inclure ce type de modifications IAM dans les chemins de code critique et haute disponibilité de votre application. Au lieu de cela, procédez aux modifications IAM dans une routine d'initialisation ou d'installation distincte que vous exécutez moins souvent.

Veillez également à vérifier que les modifications ont été propagées avant que les processus de production en dépendent. Pour plus d'informations, consultez Les modifications que j'apporte ne sont pas toujours visibles immédiatement.

Gratuité

AWS Identity and Access Management (IAM) et AWS Security Token Service (AWS STS) sont des fonctionnalités de votre AWS compte proposées sans frais supplémentaires. Vous êtes facturé uniquement lorsque vous accédez à d'autres AWS services à l'aide de vos utilisateurs IAM ou de vos informations d'identification de sécurité AWS STS temporaires. Pour plus d'informations sur les prix des autres AWS produits, consultez la page de tarification d'Amazon Web Services.

Accéder à IAM

Vous pouvez travailler avec AWS Identity and Access Management l'une des méthodes suivantes.

AWS Management Console

La console est une interface basée sur un navigateur permettant de gérer l'IAM et les ressources. AWS Pour plus d'informations sur l'accès à IAM via la console, consultez Comment se connecter à AWS dans le Guide de l'utilisateur Connexion à AWS .

AWS Outils de ligne de commande

Vous pouvez utiliser les outils de ligne de AWS commande pour émettre des commandes sur la ligne de commande de votre système afin d'exécuter l'IAM et AWS des tâches. L'utilisation de la ligne de commande peut être plus rapide et plus pratique que de la console. Les outils de ligne de commande sont également utiles si vous souhaitez créer des scripts qui exécutent AWS des tâches.

AWS fournit deux ensembles d'outils de ligne de commande : le AWS Command Line Interface(AWS CLI) et le AWS Tools for Windows PowerShell. Pour plus d'informations sur l'installation et l'utilisation du AWS CLI, consultez le guide de AWS Command Line Interface l'utilisateur. Pour plus d'informations sur l'installation et l'utilisation des outils pour Windows PowerShell, consultez le guide de AWS Tools for Windows PowerShell l'utilisateur.

Une fois connecté à la console, vous pouvez l'utiliser AWS CloudShell depuis votre navigateur pour exécuter des commandes CLI ou SDK. Les autorisations d'accès aux AWS ressources sont basées sur les informations d'identification que vous avez utilisées pour vous connecter à

la console. Selon votre expérience, vous trouverez peut-être que la CLI est une méthode plus efficace pour gérer votre Compte AWS. Pour plus d'informations, consultez Utilisation AWS CloudShell pour travailler avec AWS Identity and Access Management.

AWS SDK

AWS fournit des SDK (kits de développement logiciel) composés de bibliothèques et d'exemples de code pour différents langages de programmation et plateformes (Java, Python, Ruby, .NET, iOS, Android, etc.). Les SDK constituent un moyen pratique de créer un accès programmatique à IAM et. AWS Par exemple, ils automatisent les tâches telles que la signature cryptographique des demandes, la gestion des erreurs et les nouvelles tentatives automatiques de demande. Pour plus d'informations sur AWS les SDK, notamment sur la façon de les télécharger et de les installer, consultez la page Outils pour Amazon Web Services.

API Query IAM

Vous pouvez accéder à IAM et par AWS programmation à l'aide de l'API de requête IAM, qui vous permet d'envoyer des requêtes HTTPS directement au service. Lorsque vous utilisez l'API Query , vous devez inclure un code pour signer numériquement les demandes à l'aide de vos informations d'identification. Pour plus d'informations sur les groupes d'utilisateurs, veuillez consulter Appel de l'API IAM à l'aide de requêtes HTTP et la référence API IAM.

Quand utiliser IAM ?

Lorsque vous effectuez différentes activités professionnelles

AWS Identity and Access Management est un service d'infrastructure de base qui constitue la base du contrôle d'accès basé sur les identités internes AWS. Vous utilisez IAM chaque fois que vous accédez à votre compte AWS .

Votre utilisation d'IAM diffère selon le travail que vous effectuez dans AWS.

- Utilisateur du service – Si vous utilisez un service AWS pour effectuer votre travail, votre administrateur vous fournit les informations d'identification et les autorisations dont vous avez besoin. Plus vous utilisez de fonctions avancées pour effectuer votre travail, plus vous pouvez avoir besoin d'autorisations supplémentaires. En comprenant bien la gestion des accès, vous saurez demander les autorisations appropriées à votre administrateur.
- Administrateur de service — Si vous êtes responsable d'une AWS ressource au sein de votre entreprise, vous avez probablement un accès complet à IAM. Votre responsabilité consiste à

déterminer les fonctionnalités IAM ainsi que les ressources auxquelles les utilisateurs de votre service doivent accéder. Vous devez ensuite soumettre les demandes à votre administrateur IAM pour modifier les autorisations des utilisateurs de votre service. Consultez les informations sur cette page pour comprendre les concepts de base d'IAM.

- Administrateur IAM – Si vous êtes administrateur IAM, vous gérez les identités IAM et rédigez des politiques pour gérer l'accès à IAM.

Lorsque vous êtes autorisé à accéder aux ressources AWS

L'authentification est la façon dont vous vous connectez à AWS l'aide de vos informations d'identification. Vous devez être authentifié (connecté à AWS) en tant qu'utilisateur IAM ou en assumant un rôle IAM. Utilisateur racine d'un compte AWS

Vous pouvez vous connecter en AWS tant qu'identité fédérée en utilisant les informations d'identification fournies par le biais d'une source d'identité. AWS IAM Identity Center Les utilisateurs (IAM Identity Center), l'authentification unique de votre entreprise et vos informations d'identification Google ou Facebook sont des exemples d'identités fédérées. Lorsque vous vous connectez avec une identité fédérée, votre administrateur aura précédemment configuré une fédération d'identités avec des rôles IAM. Lorsque vous accédez à AWS l'aide de la fédération, vous assumez indirectement un rôle.

Selon le type d'utilisateur que vous êtes, vous pouvez vous connecter au portail AWS Management Console ou au portail AWS d'accès. Pour plus d'informations sur la connexion à AWS, consultez Comment vous connecter à votre compte Compte AWS dans le guide de Connexion à AWS l'utilisateur.

Si vous y accédez AWS par programmation, AWS fournit un kit de développement logiciel (SDK) et une interface de ligne de commande (CLI) pour signer cryptographiquement vos demandes à l'aide de vos informations d'identification. Si vous n'utilisez pas d' AWS outils, vous devez signer vous-même les demandes. Pour plus d'informations sur l'utilisation de la méthode recommandée pour signer vous-même les demandes, consultez la section Signature des demandes AWS d'API dans le guide de l'utilisateur IAM.

Quelle que soit la méthode d'authentification que vous utilisez, vous devrez peut-être fournir des informations de sécurité supplémentaires. Par exemple, il vous AWS recommande d'utiliser l'authentification multifactorielle (MFA) pour renforcer la sécurité de votre compte. Pour en savoir plus, consultez Authentification multifactorielle dans le Guide de l'utilisateur AWS IAM Identity Center

et Utilisation de l’authentification multifactorielle (MFA) dans l’interface AWS dans le Guide de l’utilisateur IAM.

Lorsque vous vous connectez en tant qu'utilisateur IAM

Un utilisateur IAM est une identité au sein de vous Compte AWS qui possède des autorisations spécifiques pour une seule personne ou application. Dans la mesure du possible, nous vous recommandons de vous appuyer sur des informations d’identification temporaires plutôt que de créer des utilisateurs IAM ayant des informations d’identification à long terme tels que les clés d’accès. Toutefois, si certains cas d’utilisation spécifiques nécessitent des informations d’identification à long terme avec les utilisateurs IAM, nous vous recommandons de faire pivoter les clés d’accès. Pour plus d’informations, consultez Rotation régulière des clés d’accès pour les cas d’utilisation nécessitant des informations d’identification dans le Guide de l’utilisateur IAM.

Un groupe IAM est une identité qui concerne un ensemble d’utilisateurs IAM. Vous ne pouvez pas vous connecter en tant que groupe. Vous pouvez utiliser les groupes pour spécifier des autorisations pour plusieurs utilisateurs à la fois. Les groupes permettent de gérer plus facilement les autorisations pour de grands ensembles d’utilisateurs. Par exemple, vous pouvez avoir un groupe nommé IAMAdmins et accorder à ce groupe les autorisations d’administrer des ressources IAM.

Les utilisateurs sont différents des rôles. Un utilisateur est associé de manière unique à une personne ou une application, alors qu’un rôle est conçu pour être endossé par tout utilisateur qui en a besoin. Les utilisateurs disposent d’informations d’identification permanentes, mais les rôles fournissent des informations d’identification temporaires. Pour en savoir plus, consultez Quand créer un utilisateur IAM (au lieu d’un rôle) dans le Guide de l’utilisateur IAM.

Lorsque vous assumez un rôle IAM

Un rôle IAM est une identité au sein de vous Compte AWS dotée d'autorisations spécifiques. Le concept ressemble à celui d’utilisateur IAM, mais le rôle IAM n’est pas associé à une personne en particulier. Vous pouvez assumer temporairement un rôle IAM dans le en AWS Management Console changeant de rôle. Vous pouvez assumer un rôle en appelant une opération d' AWS API AWS CLI ou en utilisant une URL personnalisée. Pour plus d’informations sur les méthodes d’utilisation des rôles, consultez Utilisation de rôles IAM dans le Guide de l’utilisateur IAM.

Les rôles IAM avec des informations d’identification temporaires sont utiles dans les cas suivants :

- Accès utilisateur fédéré – Pour attribuer des autorisations à une identité fédérée, vous créez un rôle et définissez des autorisations pour le rôle. Quand une identité externe s’authentifie,

l'identité est associée au rôle et reçoit les autorisations qui sont définies par celui-ci. Pour obtenir des informations sur les rôles pour la fédération, consultez Création d'un rôle pour un fournisseur d'identité tiers (fédération) dans le Guide de l'utilisateur IAM. Si vous utilisez IAM Identity Center, vous configurez un jeu d'autorisations. IAM Identity Center met en corrélation le jeu d'autorisations avec un rôle dans IAM afin de contrôler à quoi vos identités peuvent accéder après leur authentification. Pour plus d'informations sur les jeux d'autorisations, consultez la rubrique Jeux d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .

- Autorisations d'utilisateur IAM temporaires : un rôle ou un utilisateur IAM peut endosser un rôle IAM pour profiter temporairement d'autorisations différentes pour une tâche spécifique.
- Accès intercompte : vous pouvez utiliser un rôle IAM pour permettre à un utilisateur (principal de confiance) d'un compte différent d'accéder aux ressources de votre compte. Les rôles constituent le principal moyen d'accorder l'accès intercompte. Toutefois, dans certains Services AWS cas, vous pouvez associer une politique directement à une ressource (au lieu d'utiliser un rôle comme proxy). Pour en savoir plus sur la différence entre les rôles et les politiques basées sur les ressources pour l'accès intercompte, consultez Différence entre les rôles IAM et les politiques basées sur les ressources dans le Guide de l'utilisateur IAM.
- Accès multiservices — Certains Services AWS utilisent des fonctionnalités dans d'autres Services AWS. Par exemple, lorsque vous effectuez un appel dans un service, il est courant que ce service exécute des applications dans Amazon EC2 ou stocke des objets dans Amazon S3. Un service peut le faire en utilisant les autorisations d'appel du principal, un rôle de service ou un rôle lié au service.
  - Sessions d'accès direct (FAS) : lorsque vous utilisez un utilisateur ou un rôle IAM pour effectuer des actions AWS, vous êtes considéré comme un mandant. Lorsque vous utilisez certains services, vous pouvez effectuer une action qui initie une autre action dans un autre service. FAS utilise les autorisations du principal appelant et Service AWS, associées Service AWS à la demande, pour adresser des demandes aux services en aval. Les demandes FAS ne sont effectuées que lorsqu'un service reçoit une demande qui nécessite des interactions avec d'autres personnes Services AWS ou des ressources pour être traitée. Dans ce cas, vous devez disposer d'autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur la politique relative à la transmission de demandes FAS, consultez Sessions de transmission d'accès.
  - Rôle de service : il s'agit d'un rôle IAM attribué à un service afin de réaliser des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer une fonction du service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.

  - Rôle lié à un service — Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés au service apparaissent dans votre Compte AWS fichier et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.
- Applications exécutées sur Amazon EC2 : vous pouvez utiliser un rôle IAM pour gérer les informations d'identification temporaires pour les applications qui s'exécutent sur une instance EC2 et qui envoient des demandes d'API. AWS CLI AWS Cette solution est préférable au stockage des clés d'accès au sein de l'instance EC2. Pour attribuer un AWS rôle à une instance EC2 et le mettre à la disposition de toutes ses applications, vous devez créer un profil d'instance attaché à l'instance. Un profil d'instance contient le rôle et permet aux programmes qui s'exécutent sur l'instance EC2 d'obtenir des informations d'identification temporaires. Pour plus d'informations, consultez Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2 dans le Guide de l'utilisateur IAM.

Pour savoir dans quel cas utiliser des rôles ou des utilisateurs IAM, consultez Quand créer un rôle IAM (au lieu d'un utilisateur) dans le Guide de l'utilisateur IAM.

Lorsque vous créez des politiques et des autorisations

Vous accordez des autorisations à un utilisateur en créant une stratégie, qui est un document qui répertorie les actions qu'un utilisateur peut effectuer et les ressources que ces actions peuvent concerner. Les actions ou ressources qui ne sont pas explicitement autorisées sont refusées par défaut. Vous pouvez créer des politiques et les attacher à des principaux (utilisateurs, groupes d'utilisateurs, rôles assumés par des utilisateurs et ressources).

Ces politiques sont utilisées avec un rôle IAM :

- Politique d'approbation – Définit quels principaux peuvent assumer ce rôle et dans quelles conditions. Une politique d'approbation est un type de politique basée sur les ressources pour les rôles IAM. Un rôle ne peut disposer que d'une seule politique d'approbation.
- Politiques basées sur l'identité (en ligne et gérées) – Ces politiques définissent les autorisations que l'utilisateur du rôle est en mesure d'exécuter (ou qui lui sont refusées), et sur quelles ressources.

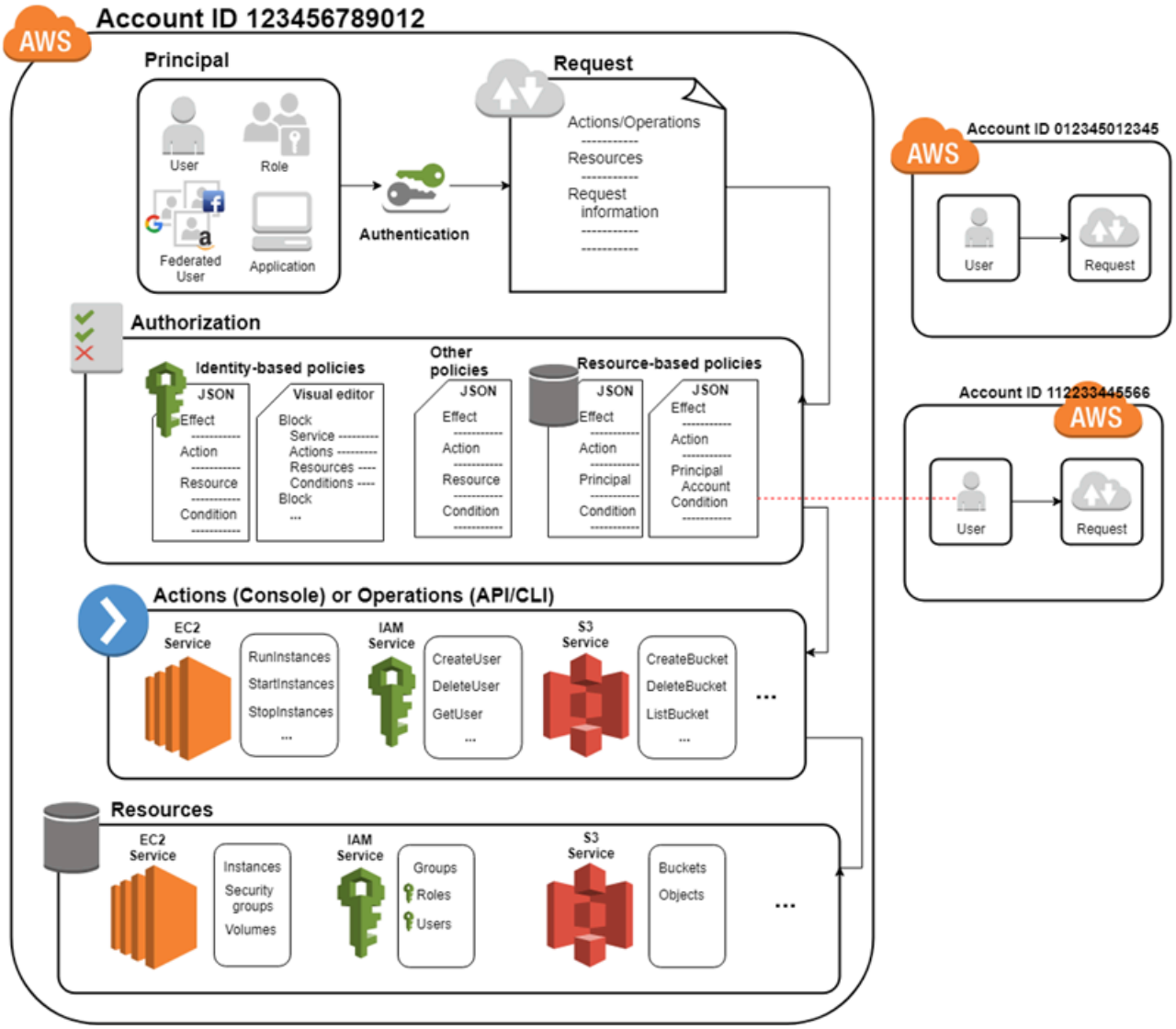
Utilisez les Exemples de politiques basées sur l'identité IAM pour vous aider à définir des autorisations pour vos identités IAM. Une fois la politique recherchée trouvée, choisissez Afficher la

politique pour afficher le JSON de la politique. Vous pouvez utiliser le document de politique JSON sous forme de modèle pour vos propres politiques.

> **Note**
>
> Si vous utilisez IAM Identity Center pour gérer vos utilisateurs, vous attribuez des jeux d'autorisations dans IAM Identity Center au lieu d'associer une politique d'autorisations à un principal. Lorsque vous attribuez un ensemble d'autorisations à un groupe ou à un autre utilisateur dans le centre d'identité AWS IAM, IAM Identity Center crée les rôles IAM correspondants dans chaque compte et associe les politiques spécifiées dans le jeu d'autorisations à ces rôles. IAM Identity Center gère le rôle et permet aux utilisateurs autorisés que vous avez définis d'assumer ce rôle. Si vous modifiez le jeu d'autorisations, IAM Identity Center s'assure que les politiques et les rôles IAM correspondants sont mis à jour en conséquence.
> Pour plus d'informations sur IAM Identity Center, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Fonctionnement de IAM

IAM fournit l'infrastructure nécessaire au contrôle de l'authentification et des autorisations de votre compte Compte AWS. L'infrastructure IAM est illustrée par le schéma suivant :



Tout d'abord, un utilisateur humain ou une application utilise ses informations de connexion pour s'authentifier auprès d' AWS. L'authentification fonctionne en faisant correspondre les informations de connexion à un principal (un utilisateur IAM, un utilisateur fédéré, un rôle IAM ou une application) approuvé par le Compte AWS.

Ensuite, une demande est effectuée pour accorder au principal l'accès aux ressources. L'accès est accordé en réponse à une demande d'autorisation. Par exemple, lorsque vous vous connectez à la console pour la première fois et que vous vous trouvez sur sa page d'accueil, vous n'accédez à aucun service spécifique. Lorsque vous sélectionnez un service, la demande d'autorisation est envoyée à ce service et celui-ci vérifie si votre identité figure sur la liste des utilisateurs autorisés, les stratégies appliquées pour contrôler le niveau d'accès accordé et toutes les autres stratégies

susceptibles d'être en vigueur. Les demandes d'autorisation peuvent être faites par des mandants au sein de vous Compte AWS ou par une autre personne en Compte AWS qui vous avez confiance.

Une fois autorisé, le principal peut prendre des mesures ou effectuer des opérations sur les ressources de votre Compte AWS. Par exemple, le principal peut lancer une nouvelle Amazon Elastic Compute Cloud instance, modifier l'appartenance à un groupe IAM ou supprimer des Amazon Simple Storage Service buckets.

Concepts de base

- Conditions
- Principal
- Demande
- Authentification
- Autorisation
- Actions ou opérations
- Ressources

Conditions

Ces termes IAM sont couramment utilisés lorsque vous travaillez avec AWS :

Ressource IAM

Les ressources IAM sont stockées dans IAM. Vous pouvez les ajouter, les modifier et les supprimer dans IAM.

- utilisateur
- groupe
- rôle
- stratégie
- objet fournisseur d'identité

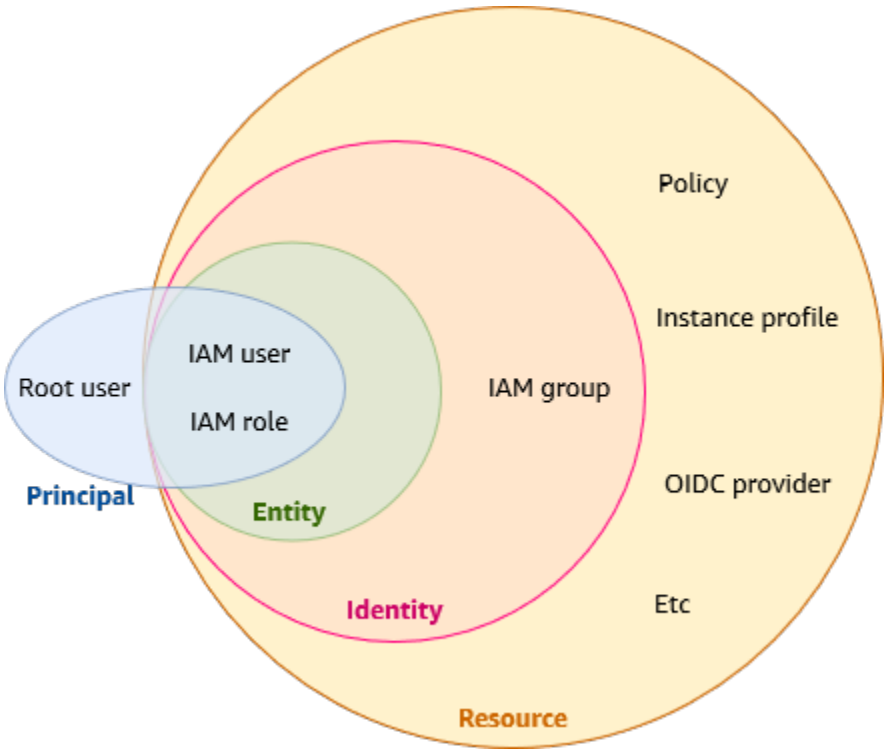
Entité IAM

Ressources IAM AWS utilisées pour l'authentification. Les entités peuvent être spécifiées en tant que principal dans une politique basée sur les ressources.

- utilisateur
- rôle

Identité IAM

Une ressource IAM qui peut être autorisée dans des politiques pour effectuer des actions et accéder aux ressources. Les identités incluent les utilisateurs, les groupes et les rôles.



Principaux

Personne ou application qui utilise le rôle Utilisateur racine d'un compte AWS, un utilisateur IAM ou un rôle IAM pour se connecter et envoyer des demandes à. AWS Les principaux incluent des utilisateurs fédérés et des rôles endossés.

Utilisateurs humains

Aussi connus sous le nom identités humaines : les personnes, les administrateurs, les développeurs, les opérateurs et les consommateurs de vos applications.

Charge de travail

Un ensemble de ressources et de code qui fournit une valeur business, par exemple une application destinée au client ou un processus de backend. Peut inclure des applications, des outils opérationnels et des composants.

Principal

Un mandant est un utilisateur humain ou une charge de travail qui peut demander une action ou une opération sur une AWS ressource. Après l'authentification, le principal peut se voir attribuer des informations d'identification permanentes ou temporaires auxquelles il peut adresser des demandes AWS, selon le type de principal. Les utilisateurs IAM et l'utilisateur root se voient attribuer des informations d'identification permanentes, tandis que les rôles des informations d'identification temporaires. En tant que bonne pratique, nous vous recommandons de demander aux utilisateurs humains et aux charges de travail d'accéder aux AWS ressources à l'aide d'informations d'identification temporaires.

Demande

Lorsqu'un principal essaie d'utiliser l' AWS Management Console AWS API, ou le AWS CLI, ce principal envoie une demande à AWS. La demande inclut les informations suivants :

- Actions ou opérations : les actions ou opérations que le principal souhaite exécuter. Il peut s'agir d'une action dans AWS Management Console ou d'une opération dans l' AWS API AWS CLI or.
- Ressources : objet de AWS ressource sur lequel les actions ou opérations sont effectuées.
- Principal : personne ou application qui utilise une entité (utilisateur ou rôle) pour envoyer la demande. Les informations sur le principal incluent les politiques associées à l'entité utilisée par le principal pour se connecter.
- Données d'environnement : informations sur l'adresse IP, l'agent utilisateur, le statut SSL ou le moment de la journée.
- Données de ressources : données liées à la ressource qui est demandée. Par exemple, ces informations peuvent inclure le nom d'une table DynamoDB ou une balise sur une instance Amazon EC2.

AWS rassemble les informations de la demande dans un contexte de demande, qui est utilisé pour évaluer et autoriser la demande.

Authentification

Un principal doit être authentifié (connecté à AWS) à l'aide de ses informations d'identification pour envoyer une demande à AWS. Certains services, tels qu'Amazon S3 et Amazon AWS STS, autorisent quelques demandes d'utilisateurs anonymes. Cependant, ils sont l'exception à la règle.

Pour vous authentifier à partir de la console en tant que utilisateur racine, vous devez vous connecter avec votre adresse e-mail et votre mot de passe. En tant qu'utilisateur fédéré, vous êtes authentifié par votre fournisseur d'identité et vous pouvez accéder aux AWS ressources en assumant des rôles IAM. En tant qu'utilisateur IAM, indiquez votre ID de compte ou alias, puis votre nom d'utilisateur et votre mot de passe. Pour authentifier les charges de travail à partir de l'API ou de la AWS CLI, vous pouvez utiliser des informations d'identification temporaires en vous voyant attribuer un rôle ou vous pouvez utiliser des informations d'identification à long terme en fournissant votre clé d'accès et votre clé secrète. Vous devrez peut-être également fournir des informations de sécurité supplémentaires. Il est AWS recommandé d'utiliser l'authentification multifactorielle (MFA) et des informations d'identification temporaires pour renforcer la sécurité de votre compte. Pour en savoir plus sur les entités IAM qui AWS peuvent s'authentifier, consultez Utilisateurs IAM et. Rôles IAM

Autorisation

Vous devez également être autorisé à terminer votre demande. Lors de l'autorisation, AWS utilise les valeurs du contexte de la demande pour rechercher les politiques qui s'appliquent à la demande. Ensuite, il utilise les politiques pour déterminer s'il autorise ou refuse la demande. La plupart des politiques sont stockées AWS sous forme de documents JSON et spécifient les autorisations pour les entités principales. Il existe plusieurs types de politiques pouvant affecter l'autorisation d'une demande. Pour autoriser vos utilisateurs à accéder aux AWS ressources de leur propre compte, vous n'avez besoin que de politiques basées sur l'identité. Les politiques basées sur les ressources sont couramment utilisées pour accorder un accès entre comptes. Les autres types de politique consistent en des fonctionnalités avancées et doivent être utilisés avec précaution.

AWS vérifie chaque politique qui s'applique au contexte de votre demande. Si une seule politique d'autorisation inclut une action refusée, AWS refuse l'intégralité de la demande et arrête l'évaluation. Ceci est appelé un refus explicite. Les demandes étant refusées par défaut, n' AWS autorise votre demande que si chaque partie de votre demande est autorisée par les politiques d'autorisation applicables. La logique d'évaluation pour une demande au sein d'un même compte utilise les règles suivantes :

- Par défaut, toutes les demandes sont refusées. (En général, les demandes effectuées à l'aide des informations d'identification du Utilisateur racine d'un compte AWS pour les ressources de ce compte sont toujours autorisées.)
- Une autorisation explicite dans une politique d'autorisations (basée sur l'identité ou les ressources) remplace cette valeur par défaut.

- L'existence d'une politique de contrôle de service Organisations, de limites d'autorisations IAM ou d'une politique de session remplace l'autorisation. S'il existe une ou plusieurs de ces sortes de politiques, elles doivent toutes autoriser la demande. Sinon, elle est refusée implicitement.
- Un refus explicite dans n'importe quelle politique remplace toutes les autorisations.

Pour en savoir plus sur la façon dont tous les types de politiques sont évalués, veuillez consulter Logique d'évaluation de politiques. Si vous avez besoin d'effectuer une demande dans un autre compte, une politique dans l'autre compte doit vous autoriser à accéder à la ressource et l'entité IAM que vous utilisez pour effectuer la demande doit avoir une politique basée sur une identité qui autorise la demande.

Actions ou opérations

Une fois que votre demande a été authentifiée et autorisée, AWS approuve les actions ou opérations figurant dans votre demande. Les opérations sont définies par un service et incluent les actions que vous pouvez exécuter sur une ressource, comme son affichage, sa création, sa modification et sa suppression. Par exemple, IAM prend en charge environ 40 actions pour une ressource utilisateur, y compris les suivantes :

- `CreateUser`
- `DeleteUser`
- `GetUser`
- `UpdateUser`

Pour autoriser un principal à exécuter une opération, vous devez inclure les actions nécessaires dans une politique qui s'applique au principal ou à la ressource affectée. Pour consulter la liste des actions, des types de ressources et des clés de condition pris en charge par chaque service, consultez la section Actions, ressources et clés de condition pour les AWS services.

Ressources

Une AWS fois les opérations de votre demande approuvées, elles peuvent être effectuées sur les ressources associées de votre compte. Une ressource est une objet existant au sein d'un service. Il peut s'agir, par exemple, d'une instance Amazon EC2, d'un utilisateur IAM et d'un compartiment Amazon S3. Le service définit un ensemble d'actions pouvant être effectuées sur chaque ressource. Si vous créez une requête pour effectuer une action non connexe sur une ressource, cette requête

est rejetée. Par exemple, si vous demandez la suppression d'un rôle IAM mais fournissez une ressource de groupe IAM, la requête échoue. Pour consulter les tables des AWS services qui identifient les ressources affectées par une action, voir Actions, ressources et clés de condition pour les AWS services.

Vue d'ensemble de la gestion des AWS identités : utilisateurs

Vous pouvez donner accès à votre compte Compte AWS à des utilisateurs spécifiques et leur fournir des autorisations spécifiques pour accéder aux ressources de votre Compte AWS. Vous pouvez utiliser à la fois IAM et AWS IAM Identity Center créer de nouveaux utilisateurs ou fédérer des utilisateurs existants dans. AWS La principale différence entre les deux est que les utilisateurs IAM reçoivent des informations d'identification à long terme pour vos AWS ressources, tandis que les utilisateurs d'IAM Identity Center disposent d'informations d'identification temporaires qui sont établies chaque fois qu'ils se connectent à. AWS Il est recommandé d'obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires plutôt qu'en tant qu'utilisateur IAM. L'une des principales utilisations pour les utilisateurs IAM est de donner aux charges de travail qui ne peuvent pas utiliser de rôles IAM la capacité d'envoyer des demandes programmatiques aux AWS services à l'aide de l'API ou de la CLI.

Rubriques

- Premier accès uniquement : vos informations d'identification utilisateur racine
- Utilisateurs IAM et utilisateurs dans IAM Identity Center
- Fédération d'utilisateurs existants
- Méthodes de contrôle d'accès

Premier accès uniquement : vos informations d'identification utilisateur racine

Lorsque vous créez un Compte AWS, vous commencez par une identité de connexion unique qui donne un accès complet à toutes Services AWS les ressources du compte. Cette identité est appelée utilisateur Compte AWS root et est accessible en vous connectant avec l'adresse e-mail et le mot de passe que vous avez utilisés pour créer le compte. Il est vivement recommandé de ne pas utiliser l’utilisateur racine pour vos tâches quotidiennes. Protégez vos informations d’identification d’utilisateur racine et utilisez-les pour effectuer les tâches que seul l’utilisateur racine peut effectuer. Pour obtenir la liste complète des tâches qui vous imposent de vous connecter en tant qu'utilisateur root, veuillez consulter Tâches nécessitant les informations d'identification de l'utilisateur root dans

le Guide de l'utilisateur IAM. Seules les politiques de contrôle des services (SCP, Service Control Policies) dans les organisations peuvent restreindre les autorisations accordées à l'utilisateur racine.

Utilisateurs IAM et utilisateurs dans IAM Identity Center

Les utilisateurs IAM ne sont pas des comptes distincts ; ce sont des utilisateurs présents dans votre compte. Chaque utilisateur dispose de son propre mot de passe pour accéder à l' AWS Management Console. Vous pouvez également créer une clé d'accès individuelle pour chaque utilisateur afin de lui permettre d'effectuer des demandes par programmation en vue d'utiliser des ressources de votre compte.

Les utilisateurs IAM reçoivent des informations d'identification à long terme pour vos AWS ressources. En tant que bonne pratique, ne créez pas d'utilisateurs IAM avec des informations d'identification à long terme pour vos utilisateurs humains. Demandez plutôt à vos utilisateurs humains d'utiliser des informations d'identification temporaires lors de l'accès AWS.

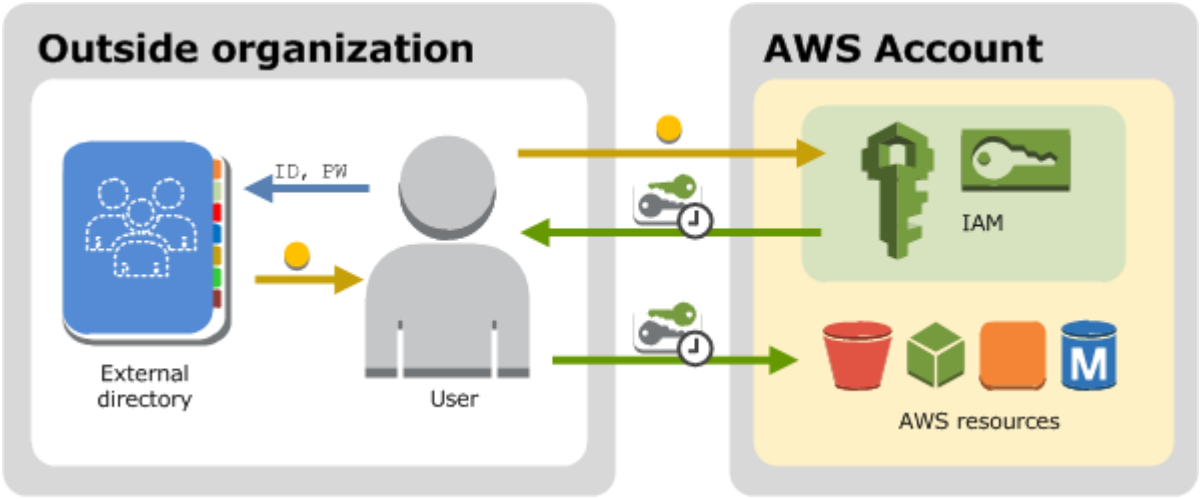
> **Note**
>
> Pour les scénarios dans lesquels vous avez besoin d'utilisateurs IAM disposant d'un accès programmatique et d'informations d'identification à long terme, nous vous recommandons de mettre à jour les clés d'accès. Pour plus d’informations, consultez Mise à jour des clés d'accès.

En revanche, utilisateurs dans IAM Identity Center des informations d'identification à court terme sont accordées à vos AWS ressources. Pour une gestion centralisée des accès, nous vous recommandons d'utiliser AWS IAM Identity Center (IAM Identity Center) pour gérer l'accès à vos comptes et les autorisations au sein de ceux-ci. IAM Identity Center est automatiquement configuré avec un répertoire Identity Center comme source d'identité par défaut, dans lequel vous pouvez créer des utilisateurs et des groupes, et attribuer leur niveau d'accès à vos AWS ressources. Pour plus d’informations, consultez Présentation de AWS IAM Identity Center dans le Guide de l’utilisateur AWS IAM Identity Center .

Fédération d'utilisateurs existants

Si les utilisateurs de votre organisation peuvent déjà s'authentifier, par exemple en se connectant à votre réseau d'entreprise, vous n'avez pas besoin de leur créer des utilisateurs IAM distincts ou des utilisateurs dans IAM Identity Center. Au lieu de cela, vous pouvez fédérer ces identités d'utilisateurs en AWS utilisant IAM ou. AWS IAM Identity Center

Le schéma suivant montre comment un utilisateur peut obtenir des informations d'identification AWS de sécurité temporaires pour accéder aux ressources de votre Compte AWS.



La fédération est particulièrement utile dans les cas suivants :

- Vos utilisateurs existent déjà dans un annuaire d'entreprise.

  Si votre annuaire d'entreprise est compatible avec le langage SAML 2.0 (Security Assertion Markup Language 2.0), vous pouvez configurer votre annuaire d'entreprise pour fournir à vos utilisateurs un accès par authentification unique (SSO). AWS Management Console Pour plus d'informations, consultez Scénarios courants d'informations d'identification temporaires.

  Si votre annuaire d'entreprise n'est pas compatible avec SAML 2.0, vous pouvez créer une application de courtier d'identité pour fournir à vos utilisateurs un accès par authentification unique (SSO). AWS Management Console Pour plus d'informations, consultez Activation de l'accès à la AWS console par un courtier d'identité personnalisé.

  Si votre annuaire d'entreprise est Microsoft Active Directory, vous pouvez l'utiliser AWS IAM Identity Center pour connecter un répertoire autogéré dans Active Directory ou un annuaire AWS Directory Servicepour établir un lien de confiance entre votre annuaire d'entreprise et votre Compte AWS.

  Si vous utilisez un fournisseur d'identité externe (IdP) tel qu'Okta ou Microsoft Entra pour gérer les utilisateurs, vous pouvez l'utiliser AWS IAM Identity Center pour établir un lien de confiance entre votre IdP et votre. Compte AWS Pour plus d'informations, consultez la section Se connecter à un fournisseur d'identité externe dans le Guide de l'utilisateur AWS IAM Identity Center .
- Vos utilisateurs disposent déjà d'identités Internet.

Si vous créez une application mobile ou une application basée sur le web permettant aux utilisateurs de s'identifier à l'aide d'un fournisseur d'entité Internet comme Login with Amazon, Facebook, Google ou tout autre fournisseur d'identité compatible avec OpenID Connect (OIDC), l'application peut utiliser la fédération pour accéder à AWS. Pour plus d'informations, consultez Fédération OIDC.

> **Tip**
>
> Pour utiliser la fédération d'identité avec des fournisseurs d'identité, nous vous recommandons d'utiliser Amazon Cognito.

Méthodes de contrôle d'accès

Voici comment contrôler l'accès à vos AWS ressources.

Type d'accès utilisateur	Pourquoi devrais-je l'utiliser ?	Comment puis-je obtenir plus d'informations ?
Accès par authentification unique pour les utilisateurs humains, tels que les utilisateurs de votre personnel, aux ressources AWS à l'aide d'IAM Identity Center	L'IAM Identity Center fournit un espace central qui regroupe l'administration des utilisateurs et leur accès aux Comptes AWS applications cloud. Vous pouvez configurer un magasin d'identités dans IAM Identity Center ou configurer une fédération avec un fournisseur d'identité (IdP) existant. Il est recommandé d'accorder à vos utilisateurs humains des informations d'identification limitées aux AWS ressources selon les besoins en tant que	Pour plus d'informations sur la configuration d'IAM Identity Center, consultez Mise en route dans le Guide de l'utilisateur AWS IAM Identity Center . Pour plus d'informations sur l'utilisation de MFA dans IAM Identity Center, consultez Authentification multifactorielle dans le Guide de l'utilisateur AWS IAM Identity Center .

Type d'accès utilisateur	Pourquoi devrais-je l'utiliser ?	Comment puis-je obtenir plus d'informations ?
	bonne pratique en matière de sécurité. Les utilisateurs bénéficient d'une expérience de connexion simplifiée et vous conservez le contrôle de leur accès aux ressources à partir d'un système unique. IAM Identity Center prend en charge l'authentification multifactorielle (MFA) pour renforcer la sécurité des comptes.	
Accès fédéré pour les utilisateurs humains, tels que les utilisateurs de votre personnel, aux services AWS utilisant des fournisseurs d'identité IAM	Les supports IAM sont IdPs compatibles avec OpenID Connect (OIDC) ou SAML 2.0 (Security Assertion Markup Language 2.0). Après avoir créé un fournisseur d'identité IAM, vous devez créer un ou plusieurs rôles IAM qui peuvent être attribués de manière dynamique à un utilisateur fédéré.	Pour plus d'informations sur les fournisseurs d'identité IAM et la fédération, consultez Fournisseurs d'identité et fédération.

Type d'accès utilisateur	Pourquoi devrais-je l'utiliser ?	Comment puis-je obtenir plus d'informations ?
Accès multicompte entre Comptes AWS	Vous souhaitez partager l'accès à certaines AWS ressources avec des utilisateurs d'autres ressources Comptes AWS. Les rôles constituent le principal moyen d’accorder l’accès intercompte. Toutefois, certains services AWS vous permettent d'attacher une politique directement à une ressource (au lieu d'utiliser un rôle en tant que proxy). On les appelle « politiques basées sur une ressource ».	Pour plus d'informations sur les rôles IAM, consultez Rôles IAM. Pour plus d’informations sur les rôles liés à un service, consultez Utilisation des rôles liés à un service. Pour connaître les services qui prennent en charge l'utilisation de rôles liés à un service, consultez AWS services qui fonctionnent avec IAM. Recherchez les services qui comportent un Oui dans la colonne Rôle lié à un service. Pour consulter la documentation relative au rôle lié à un service, sélectionnez le lien associé à Oui dans cette colonne.

Type d'accès utilisateur	Pourquoi devrais-je l'utiliser ?	Comment puis-je obtenir plus d'informations ?
Informations d'identification à long terme pour les utilisateurs IAM désignés dans votre Compte AWS	Il se peut que vous ayez des cas d'utilisation spécifiques qui nécessitent des informations d'identification à long terme avec des utilisateurs IAM intégrés. AWS Vous pouvez utiliser IAM pour créer ces utilisateurs IAM dans votre Compte AWS et utiliser IAM pour gérer leurs autorisations. Voici certaines des fonctionnalités les plus utilisées : • Charges de travail qui ne peuvent pas utiliser des rôles IAM • AWS Clients tiers nécessitant un accès programmatique via des clés d'accès • Informations d'identification spécifiques au service pour ou AWS CodeCommit Amazon Keyspaces • AWS IAM Identity Center n'est pas disponible pour votre compte et vous n'avez aucun autre fournisseur d'identité En guise de bonne pratique pour les scénarios dans lesquels vous avez besoin d'utilisateurs IAM disposant	Pour plus d'informations sur la configuration d'un utilisateur IAM, consultez Création d'un utilisateur IAM dans votre Compte AWS. Pour plus d'informations sur les clés d'accès utilisateur IAM, consultezGestion des clés d'accès pour les utilisateurs IAM. Pour plus d'informations sur les informations d'identification spécifiques à un service pour ou AWS CodeCommit Amazon Keyspaces, consultez et. Utilisation d'IAM avec CodeCommit : informations d'identification Git, clés SSH et AWS clés d'accès Utilisation d'IAM avec Amazon Keyspaces (pour Apache Cassandra)

Type d'accès utilisateur	Pourquoi devrais-je l'utiliser ?	Comment puis-je obtenir plus d'informations ?
	d'un accès programmatique et d'informations d'identification à long terme, nous vous recommandons d'effectuer une mise à jour des clés d'accès. Pour plus d'informations, consultez Mise à jour des clés d'accès.	

Présentation de la gestion des accès : autorisations et politiques

La partie de gestion des accès de AWS Identity and Access Management (IAM) vous aide à définir ce qu'une entité principale est autorisée à faire dans un compte. Une entité principal est une personne ou une application qui est authentifiée à l'aide d'une entité IAM (utilisateur ou rôle). La gestion des accès est souvent appelée autorisation. Vous gérez l'accès en AWS créant des politiques et en les associant à des identités IAM (utilisateurs, groupes d'utilisateurs ou rôles) ou à des AWS ressources. Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit leurs autorisations. AWS évalue ces politiques lorsqu'un mandant utilise une entité IAM (utilisateur ou rôle) pour faire une demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée. La plupart des politiques sont stockées AWS sous forme de documents JSON. Pour plus d'informations sur les types de politiques et les utilisations, veuillez consulter Politiques et autorisations dans IAM.

Politiques et comptes

Si vous gérez un seul compte dans AWS, vous définissez les autorisations au sein de ce compte à l'aide de politiques. Si vous gérez des autorisations sur plusieurs comptes, la gestion des autorisations pour vos utilisateurs est plus compliquée. Vous pouvez utiliser des rôles IAM, des politiques basées sur les ressources ou des listes de contrôle d'accès (ACL) pour des autorisations entre comptes. Toutefois, si vous possédez plusieurs comptes, nous vous recommandons plutôt d'utiliser le AWS Organizations service pour vous aider à gérer ces autorisations. Pour plus d'informations, voir Qu'est-ce que c'est AWS Organizations ? dans le Guide de l'utilisateur des Organizations.

Politiques et utilisateurs

Les utilisateurs IAM sont des identités dans le service. Lorsque vous créez un utilisateur IAM, il ne peut accéder à rien dans votre compte tant que vous ne lui en donnez pas l'autorisation. Vous accordez des autorisations à un utilisateur en créant une politique basée sur l'identité, qui est attachée à l'utilisateur ou à un groupe auquel l'utilisateur appartient. L'exemple suivant illustre une politique JSON qui autorise l'utilisateur à exécuter toutes les actions Amazon DynamoDB (`dynamodb:*`) sur le tableau `Books` dans le compte `123456789012` de la région `us-east-2`.

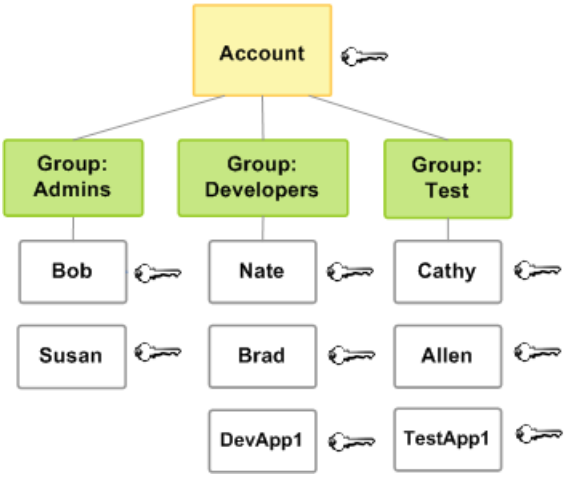
```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:123456789012:table/Books"
  }
}
```

Une fois cette politique attachée à votre utilisateur IAM, seul ce dernier dispose des autorisations DynamoDB. La plupart des utilisateurs disposent de plusieurs politiques qui une fois réunies constituent leurs autorisations.

Les actions ou ressources qui ne sont pas autorisées explicitement sont refusées par défaut. Par exemple, si la politique précédente est la seule attachée à un utilisateur, ce dernier est autorisé à exécuter des actions DynamoDB uniquement sur le tableau `Books`. Les actions sur toutes les autres tables sont interdites. De même, l'utilisateur n'est pas autorisé à effectuer des actions dans Amazon EC2, Amazon S3 ou dans tout autre AWS service. Cela s'explique car les autorisations permettant d'utiliser ces services ne sont pas incluses dans la politique.

Politiques et groupes

Vous pouvez organiser les utilisateurs IAM en groupes IAM et attacher une politique à un groupe. Dans ce cas, les utilisateurs disposent toujours de leurs propres informations d'identification, mais tous les utilisateurs d'un groupe détiennent les autorisations qui sont attachées à ce groupe. Utilisez des groupes pour faciliter la gestion des autorisations et de suivre nos Bonnes pratiques de sécurité dans IAM.



Plusieurs politiques accordant différentes autorisations peuvent être attachées à des utilisateurs ou des groupes. Dans ce cas, les autorisations des utilisateurs sont calculées sur la combinaison des politiques. Mais le principe de base s'applique toujours : si l'utilisateur n'a pas reçu une autorisation explicite pour une action et une ressource, il ne dispose pas de cette autorisation.

Utilisateurs fédérés et rôles

Les utilisateurs fédérés n'ont pas d'identité permanente comme Compte AWS les utilisateurs IAM. Pour attribuer des autorisations à des utilisateurs fédérés, vous pouvez créer une entité appelée rôle et définir des autorisations pour le rôle. Lorsqu'un utilisateur fédéré se connecte à AWS, il est associé au rôle et reçoit les autorisations définies dans le rôle. Pour plus d’informations, consultez Création d'un rôle pour un fournisseur d'identité tiers (fédération).

Politiques basées sur les ressources et politiques basées sur l'identité

Les politiques basées sur l'identité sont des politiques d'autorisations que vous pouvez attacher à une identité IAM, tel qu'un utilisateur, un groupe ou un rôle IAM. Les politiques basées sur les ressources sont des politiques d'autorisations que vous attachez à une ressource, telle qu'un compartiment Amazon S3 ou une politique d'approbation de rôle IAM.

Les politiques basées sur l'identité contrôlent les actions que peut effectuer l'identité, sur quelles ressources et dans quelles conditions. Les politiques basées l'identité peuvent être classées comme suit :

- Politiques gérées : politiques autonomes basées sur l'identité que vous pouvez associer à plusieurs utilisateurs, groupes et rôles au sein de votre. Compte AWS Vous pouvez utiliser deux types de politiques gérées :

  - AWS politiques gérées : politiques gérées créées et gérées par AWS. Si vous utilisez des politiques pour la première fois, nous vous recommandons de commencer par utiliser des politiques AWS gérées.
  - Politiques gérées par le client : politiques gérées que vous créez et gérez dans votre Compte AWS. Les politiques gérées par le client fournissent un contrôle plus précis de vos politiques que les politiques AWS gérées. Vous pouvez créer, modifier et valider une politique IAM dans l'éditeur visuel ou en créant le document de politique JSON directement. Pour plus d'informations, consultez Création de politiques IAM et Modification de politiques IAM.
- Politiques en ligne : politiques que vous créez et gérez, et qui sont intégrées directement à un utilisateur, groupe ou rôle. Dans la plupart des cas, nous vous déconseillons d'utiliser des politiques en ligne.

Les politiques basées sur les ressources contrôlent les actions qu'un principal spécifique peut effectuer sur cette ressource et dans quelles conditions. Les politiques basées sur les ressources sont des politiques en ligne et il n'y a pas de politiques basées sur des ressources gérées. Pour permettre un accès comptes multiples , vous pouvez spécifier un compte entier ou des entités IAM dans un autre compte en tant que principal dans une politique basée sur les ressources.

Le service IAM prend en charge un seul type de politique basée sur les ressources, nommé politique d'approbation de rôle, qui est attaché à un rôle IAM. Étant donné qu'un rôle IAM est à la fois une identité et une ressource qui prend en charge les politiques basées sur les ressources, vous devez associer une politique d'approbation et une politique basée sur une identité à un rôle IAM. Les politiques d'approbation définissent quelles entités principaux (comptes, utilisateurs, rôles et utilisateurs fédérés) peuvent endosser le rôle. Pour en savoir plus sur la façon dont les rôles IAM sont différents d'autres politiques basées sur les ressources, veuillez consulter Accès intercompte aux ressources dans IAM.

Pour connaître les services qui prennent en charge les politiques basées sur les ressources, voir AWS services qui fonctionnent avec IAM. Pour en savoir plus sur politiques basées sur les ressources, voir Politiques basées sur l'identité et Politiques basées sur une ressource.

À quoi sert ABAC ? AWS

Le contrôle d'accès par attributs (ABAC) est une stratégie d'autorisation qui définit des autorisations en fonction des attributs. Dans AWS, ces attributs sont appelés balises. Vous pouvez associer des balises aux ressources IAM, notamment aux entités IAM (utilisateurs ou rôles) et aux AWS

ressources. Vous pouvez créer une seule politique ABAC ou un petit nombre de politiques pour vos principaux IAM. Ces politiques ABAC sont conçues pour autoriser des opérations lorsque la balise du principal correspond à celle de la ressource. L'ABAC est utile dans les environnements qui connaissent une croissance rapide et pour les cas où la gestion des politiques devient fastidieuse.

Par exemple, vous pouvez créer trois rôles avec la clé de balise `access-project`. Définissez la valeur de la balise du premier rôle sur `Heart`, celle du deuxième sur `Star`, et celle du troisième sur `Lightning`. Vous pouvez alors utiliser une seule politique qui autorise l'accès lorsque le rôle et la ressource sont balisés avec la même valeur pour `access-project`. Pour un didacticiel détaillé expliquant comment utiliser ABAC dans AWS, voirTutoriel IAM : définir les autorisations d'accès aux AWS ressources en fonction des balises. Pour en savoir plus sur les services qui prennent en charge ABAC, consultez AWS services qui fonctionnent avec IAM.

Comparaison du contrôle ABAC et du modèle RBAC traditionnel

Le modèle d'autorisation traditionnel utilisé dans IAM est appelé contrôle d'accès basé sur les rôles (RBAC). Le RBAC accorde des droits aux utilisateurs en fonction de leur activité professionnelle, appelée rôle en dehors d' AWS . Au sein AWS d'un rôle, on entend généralement un rôle IAM, qui est une identité que vous pouvez assumer dans IAM. IAM inclut des politiques gérées pour des fonctions de tâches, qui alignent les autorisations sur une fonction de tâche dans un modèle RBAC.

Dans IAM, vous implémentez le RBAC en créant différentes politiques pour différentes activités professionnelles. Vous attachez ensuite les politiques à des identités (utilisateurs IAM, groupes d'utilisateurs ou rôles IAM). Les bonnes pratiques consistent à n'octroyer que les autorisations minimales nécessaires pour la tâche à accomplir. C'est ce qu'on appelle le principe de moindre privilège. Pour ce faire, dressez la liste des ressources spécifiques auxquelles l'activité professionnelle peut accéder. Le modèle RBAC traditionnel a pour inconvénient de nécessiter la mise à jour des politiques pour autoriser l'accès aux nouvelles ressources ajoutées par les employés.

Par exemple, supposons que vous ayez trois projets, nommés `Heart`, `Star` et `Lightning`, sur lesquels travaillent vos employés. Vous créez un rôle IAM pour chaque projet. Vous attachez ensuite des politiques à chaque rôle IAM pour définir les ressources auxquelles toute personne autorisée à endosser le rôle peut accéder. Si un employé change de poste au sein de votre entreprise, vous lui assignez un autre rôle IAM. Une personne ou un programme peut avoir plusieurs rôles. Toutefois, le projet `Star` peut nécessiter des ressources supplémentaires, telles qu'un nouveau conteneur Amazon EC2. Dans ce cas, vous devez mettre à jour la politique attachée au rôle `Star` pour spécifier la nouvelle ressource du conteneur. Sinon, les membres du projet `Star` ne seront pas autorisés à accéder au nouveau conteneur.

L'ABAC offre les avantages suivants par rapport au modèle RBAC traditionnel

- Les autorisations de l'ABAC sont évolutives. L'administrateur n'a plus besoin de mettre à jour les politiques existantes pour autoriser l'accès aux nouvelles ressources. Par exemple, supposons que vous ayez conçu votre stratégie ABAC avec la balise `access-project` Un développeur utilise le rôle avec la balise `access-project` = `Heart`. Lorsque des membres du projet `Heart` ont besoin de ressources Amazon EC2 supplémentaires, le développeur peut créer des instances Amazon EC2 avec la balise `Heart` = `access-project`. N'importe quel membre du projet `Heart` peut alors démarrer et arrêter ces instances, car leurs valeurs de balise correspondent.
- L'ABAC exige moins de politiques. Comme vous n'avez pas besoin de créer une politique pour chaque activité professionnelle, vous créez moins de politiques. Leur gestion s'en trouve simplifiée.
- Grâce à l'ABAC, les équipes peuvent évoluer et se développer rapidement. En effet, les autorisations d'accès aux nouvelles ressources sont automatiquement accordées en fonction des attributs. Par exemple, si votre entreprise prend déjà en charge les projets `Heart` et `Star` avec l'ABAC, il est facile d'ajouter un nouveau projet `Lightning`. Un administrateur IAM crée un rôle avec la balise `access-project` = `Lightning`. Il n'est pas nécessaire de modifier la politique pour prendre en charge un nouveau projet. Toute personne autorisée à endosser le rôle peut créer et afficher des instances balisées avec `access-project` = `Lightning`. En outre, un membre de l'équipe peut passer du projet `Heart` au projet `Lightning`. L'administrateur IAM attribue à l'utilisateur un rôle IAM différent. Il n'est pas nécessaire de modifier les politiques d'autorisations.
- Les autorisations détaillées sont possibles avec l'ABAC. Lorsque vous créez des politiques, la bonne pratique consiste à accorder un moindre privilège. Avec le modèle RBAC traditionnel, vous devez écrire une politique qui autorise l'accès à des ressources spécifiques uniquement. Avec l'ABAC, vous pouvez autoriser des actions sur toutes les ressources, mais uniquement si les balises de la ressource et du principal correspondent.
- Utilisez les attributs des employés figurant dans l'annuaire de votre entreprise avec l'ABAC. Vous pouvez configurer votre fournisseur SAML ou OIDC pour qu'il transmette des balises de session à. AWS Lorsque vos employés se fédèrent dans AWS, leurs attributs sont appliqués au principal qui en résulte dans AWS. Vous pouvez ensuite utiliser l'ABAC pour autoriser ou refuser des autorisations sur la base de ces attributs.

Pour un didacticiel détaillé expliquant comment utiliser ABAC dans AWS, voirTutoriel IAM : définir les autorisations d'accès aux AWS ressources en fonction des balises.

Fonctions de sécurité en dehors d'IAM

Vous utilisez IAM pour contrôler l'accès aux tâches effectuées à l'aide des outils de AWS Management Console ligne de AWS commande ou des opérations d'API de service à l'aide des AWS SDK. Certains AWS produits disposent également d'autres moyens de sécuriser leurs ressources. La liste suivante fournit quelques exemples, mais elle n'est pas exhaustive.

Amazon EC2

Dans Amazon Elastic Compute Cloud, vous vous connectez à une instance à l'aide d'une paire de clés (pour les instances Linux) ou d'un nom utilisateur et mot de passe (pour les instances Microsoft Windows).

Pour plus d'informations, consultez la documentation de suivante :

- Commencer à utiliser les instances Linux Amazon EC2 dans le guide de l'utilisateur Amazon EC2
- Commencer à utiliser les instances Windows Amazon EC2 dans le guide de l'utilisateur Amazon EC2

Amazon RDS

Dans Amazon Relational Database Service, vous vous connectez au moteur de base de données à l'aide d'un nom utilisateur et mot de passe qui sont associés à la base de données.

Pour de plus amples informations, veuillez consulter Mise en route avec Amazon RDS dans le Guide de l'utilisateur Amazon RDS.

Amazon EC2 et Amazon RDS

Dans Amazon EC2 et Amazon RDS, vous utilisez des groupes de sécurité pour contrôler le trafic d'une instance ou base de données.

Pour plus d'informations, consultez la documentation de suivante :

- Groupes de sécurité Amazon EC2 pour instances Linux dans le guide de l'utilisateur Amazon EC2
- Groupes de sécurité Amazon EC2 pour instances Windows dans le guide de l'utilisateur Amazon EC2

- Groupes de sécurité Amazon RDS dans le Guide de l'utilisateur Amazon RDS

WorkSpaces

Sur Amazon WorkSpaces, les utilisateurs se connectent à un ordinateur de bureau à l'aide d'un nom d'utilisateur et d'un mot de passe.

Pour plus d'informations, consultez Getting Started with WorkSpaces dans le guide d' WorkSpaces administration Amazon.

Amazon WorkDocs

Sur Amazon WorkDocs, les utilisateurs ont accès aux documents partagés en se connectant à l'aide d'un nom d'utilisateur et d'un mot de passe.

Pour plus d'informations, consultez Getting Started with Amazon WorkDocs dans le guide d' WorkDocs administration Amazon.

Ces méthodes de contrôle d'accès ne font pas partie d'IAM. IAM vous permet de contrôler la façon dont ces AWS produits sont administrés : création ou résiliation d'une instance Amazon EC2, configuration de nouveaux bureaux, etc. WorkSpaces Autrement dit, IAM vous aide à contrôler les tâches exécutées en effectuant des demandes à Amazon Web Services ; vous pouvez également contrôler l'accès à AWS Management Console. Cependant, IAM ne vous aide pas à gérer la sécurité pour des tâches telles que la connexion à un système d'exploitation (Amazon EC2), à une base de données (Amazon RDS), à un ordinateur de bureau (WorkSpacesAmazon) ou à un site de collaboration (Amazon). WorkDocs

Lorsque vous travaillez avec un AWS produit spécifique, assurez-vous de lire la documentation pour connaître les options de sécurité pour toutes les ressources appartenant à ce produit.

Liens rapides pour les tâches courantes

Utilisez les liens suivants pour obtenir de l'aide sur les tâches courantes associées à IAM.

Connexion pour différents types d'utilisateurs

Connectez-vous à la console IAM en choisissant IAM user et en saisissant votre Compte AWS identifiant ou votre alias de compte. Sur la page suivante, saisissez votre nom d'utilisateur IAM et votre mot de passe.

Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l‘URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l‘utilisateur IAM Identity Center.

Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

Consultez la section En quoi consiste la AWS connexion dans le Guide de l'utilisateur Connexion à AWS pour déterminer votre type d'utilisateur et votre page de connexion.

Gérer les mots de passe des utilisateurs

Vous avez besoin d'un mot de passe pour accéder au AWS Management Console, y compris pour accéder aux informations de facturation.

Pour votre Utilisateur racine d'un compte AWS, voir Modifier le mot de passe du Utilisateur racine d'un compte AWS dans le Guide AWS Account Management de référence

Pour un utilisateur IAM, veuillez consulter Gestion des mots de passe des utilisateurs IAM.

Gérer les autorisations des utilisateurs

Vous utilisez des politiques pour accorder des autorisations aux utilisateurs IAM de votre Compte AWS. Les utilisateurs IAM ne disposent d'aucune autorisation lors de leur création. Vous devez donc ajouter des autorisations pour leur permettre d'utiliser les AWS ressources.

Pour activer l’accès, ajoutez des autorisations à vos utilisateurs, groupes ou rôles :

- Utilisateurs et groupes dans AWS IAM Identity Center :

  Créez un jeu d’autorisations. Suivez les instructions de la rubrique Création d’un jeu d’autorisations du Guide de l’utilisateur AWS IAM Identity Center .
- Utilisateurs gérés dans IAM par un fournisseur d’identité :

  Créez un rôle pour la fédération d’identité. Pour plus d’informations, voir la rubrique Création d’un rôle pour un fournisseur d’identité tiers (fédération) du Guide de l’utilisateur IAM.
- Utilisateurs IAM :
  - Créez un rôle que votre utilisateur peut assumer. Suivez les instructions de la rubrique Création d’un rôle pour un utilisateur IAM du Guide de l’utilisateur IAM.

- (Non recommandé) Attachez une politique directement à un utilisateur ou ajoutez un utilisateur à un groupe d'utilisateurs. Suivez les instructions de la rubrique Ajout d'autorisations à un utilisateur (console) du Guide de l'utilisateur IAM.

Pour plus d'informations, consultez Gestion des politiques IAM.

Répertoriez les utilisateurs de votre Compte AWS compte et obtenez des informations sur leurs informations d'identification

veuillez consulter Obtenir des rapports d'informations d'identification pour votre Compte AWS.

Ajouter l'authentification multi-facteurs (MFA)

Pour ajouter un dispositif MFA virtuel, veuillez consulter l'une des rubriques suivantes :

- Activation d'un dispositif MFA virtuel pour votre Utilisateur racine d'un compte AWS (console)
- Activation d'un dispositif MFA virtuel pour un utilisateur IAM (Console)

Pour ajouter une clé de sécurité FIDO, veuillez consulter l'une des rubriques suivantes :

- Activer une clé d'accès ou une clé de sécurité pour la Utilisateur racine d'un compte AWS (console)
- Activer une clé d'accès ou une clé de sécurité pour un autre utilisateur IAM (console)

Pour ajouter un dispositif MFA matériel, veuillez consulter l'une des rubriques suivantes :

- Activer un jeton TOTP matériel pour Utilisateur racine d'un compte AWS (console).
- Activation d'un jeton TOTP matériel pour un autre utilisateur IAM (console)

Obtenir une clé d'accès

Vous pouvez utiliser une clé d'accès pour effectuer des AWS demandes à l'aide AWS des SDK, des outils de ligne de AWS commande ou des opérations d'API.

> **Important**
>
> Les bonnes pratiques consistent à utiliser des informations d'identification de sécurité temporaires (comme des rôles IAM) plutôt que de créer des informations d'identification à long terme comme des clés d'accès. Avant de créer des clés d'accès, passez en revue les alternatives aux clés d'accès à long terme.

Pour obtenir des conseils qui vous aideront à protéger vos clés d'accès, consultez la section Sécurisation des clés d'accès.

Pour plus d’informations sur la gestion des clés d’accès d’un utilisateur IAM, consultez Gestion des clés d'accès pour les utilisateurs IAM.

Pour plus d'informations sur les informations d'identification de sécurité disponibles pour vous Compte AWS, consultez la section Informations d'identification AWS de sécurité.

Baliser les ressources IAM

Vous pouvez baliser les ressources IAM suivantes :

- Utilisateurs IAM
- Rôles IAM
- Politiques gérées par le client
- Fournisseurs d’identité
- Certificats de serveur
- Appareils MFA virtuels

Pour en savoir plus sur les balises dans IAM, veuillez consulter Balisage des ressources IAM.

Pour en savoir plus sur l'utilisation de balises pour contrôler l'accès aux AWS ressources, consultezContrôle de l'accès aux AWS ressources à l'aide de balises.

Afficher les actions, les ressources et les clés de condition pour tous les services

Cet ensemble de documentation de référence peut vous aider à rédiger des politiques IAM détaillées. Chaque service AWS définit les actions, les ressources et les clés de contexte de condition que vous utilisez dans les politiques IAM. Pour en savoir plus, consultez Actions, ressources et clés de condition pour les AWS services.

Commencez avec tous AWS

Cet ensemble de documentation traite principalement du service IAM. Pour en savoir plus sur la prise en main AWS et l'utilisation de plusieurs services pour résoudre un problème tel que la création et le lancement de votre premier projet, consultez le Centre de ressources pour la mise en route.

Fonction de recherche de la console IAM

N'hésitez pas à utiliser la page de recherche de la console IAM car il s'agit d'une option plus rapide pour trouver des ressources IAM. Vous pouvez utiliser la recherche dans la console pour localiser les

clés d'accès associées à votre compte, aux entités IAM (telles que les utilisateurs, les groupes, les rôles, les fournisseurs d'identité), aux politiques par nom, etc.

La fonction de recherche de la console IAM vous permet de rechercher tous les éléments suivants :

- Les noms d'entités IAM qui correspondent à vos mots-clés (pour des utilisateurs, des groupes, des rôles, des fournisseurs d'identité et des politiques)
- Les tâches qui correspondent à vos mots-clés

La fonction de recherche de la console IAM ne renvoie pas d'informations sur IAM Access Analyzer.

Chaque ligne des résultats de la recherche est un lien actif. Par exemple, vous pouvez choisir le nom de l'utilisateur dans les résultats de la recherche, ce qui vous dirigera vers la page détaillée de cet utilisateur. Vous pouvez également choisir un lien d'action, par exemple Créer un utilisateur, pour accéder à la page Créer un utilisateur.

> **Note**
>
> La recherche de clé d'accès nécessite que vous saisissiez l'ID de clé d'accès complet dans la zone de recherche. Le résultat de la recherche affiche l'utilisateur associé à cette clé. À partir de là, vous pouvez accéder directement à la page de cet utilisateur, et même y gérer sa clé d'accès.

Utilisation de la fonction de recherche de la console IAM

Utilisez la page Search (Recherche) de la console IAM pour rechercher des éléments associés à ce compte.

Pour rechercher des éléments sur la console IAM

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d'accueil de la console, sélectionnez le service IAM.
3. Dans le panneau de navigation, sélectionnez Search (Recherche).
4. Dans la zone Recherche, saisissez vos mots-clés de recherche.
5. Choisissez un lien dans la liste des résultats de la recherche pour accéder à la partie correspondante de la console.

Icônes dans les résultats de recherche de la console IAM

Les icônes suivantes identifient les types d'éléments obtenus par une recherche :

Icône	Description
	Utilisateurs IAM
	Groupes IAM
	Rôles IAM
	Politiques IAM
	Tâches comme « créer un utilisateur » ou « attacher une politique »
	Résultats obtenus avec le mot-clé **`delete`**

Exemples d'expressions de recherche

Vous pouvez utiliser les expressions suivantes dans la recherche IAM. Remplacez les termes en italique par les noms des utilisateurs, groupes, rôles, clés d'accès, politiques ou fournisseurs d'identité IAM que vous souhaitez rechercher.

- ***`user_name`*** ou ***`group_name`*** ou ***`role_name`*** ou ***`policy_name`*** ou ***`identity_provider_name`***
- ***`access_key`***
- **`add user`** ***`user_name`*** **`to groups`** ou **`add users to group`** ***`group_name`***
- **`remove user`** ***`user_name`*** **`from groups`**
- **`delete`** ***`user_name`*** ou **`delete`** ***`group_name`*** ou **`delete`** ***`role_name`*** ou **`delete`** ***`policy_name`*** ou **`delete`** ***`identity_provider_name`***
- **`manage access keys`** ***`user_name`***

- **manage signing certificates *user_name***
- **users**
- **manage MFA for *user_name***
- **manage password for *user_name***
- **create role**
- **password policy**
- **edit trust policy for role *role_name***
- **show policy document for role *role_name***
- **attach policy to *role_name***
- **create managed policy**
- **create user**
- **create group**
- **attach policy to *group_name***
- **attach entities to *policy_name***
- **detach entities from *policy_name***

Création de AWS Identity and Access Management ressources avec AWS CloudFormation

AWS Identity and Access Management est intégré à AWS CloudFormation un service qui vous aide à modéliser et à configurer vos AWS ressources afin que vous puissiez passer moins de temps à créer et à gérer vos ressources et votre infrastructure. Vous créez un modèle qui décrit toutes les AWS ressources souhaitées (telles que les clés d'accès, les groupes, les politiques de groupe, les profils d'instance, les politiques gérées, les fournisseurs OIDC, les politiques intégrées, les rôles, les politiques de rôle, les fournisseurs SAML, les certificats de serveur, les rôles liés aux services, les utilisateurs (et l'ajout d'utilisateurs aux groupes), les politiques utilisateur et les dispositifs MFA virtuels), puis vous AWS CloudFormation approvisionnez et configurez ces ressources pour vous.

Lorsque vous l'utilisez AWS CloudFormation, vous pouvez réutiliser votre modèle pour configurer vos ressources IAM de manière cohérente et répétée. Décrivez vos ressources une seule fois, puis distribuez les mêmes ressources encore et encore dans plusieurs Comptes AWS régions.

IAM et modèles AWS CloudFormation

Pour fournir et configurer des ressources pour IAM et les services associés, vous devez comprendre les AWS CloudFormation modèles. Les modèles sont des fichiers texte formatés en JSON ou YAML. Ces modèles décrivent les ressources que vous souhaitez mettre à disposition dans vos AWS CloudFormation piles. Si vous n'êtes pas familiarisé avec JSON ou YAML, vous pouvez utiliser AWS CloudFormation Designer pour vous aider à démarrer avec les AWS CloudFormation modèles. Pour plus d'informations, consultez Qu'est-ce que AWS CloudFormation Designer ? dans le AWS CloudFormation Guide de l'utilisateur.

IAM prend en charge la création de clés d'accès, de groupes, de politiques de groupe, de profils d'instance, de politiques gérées, de fournisseurs OIDC, de politiques intégrées, de rôles, de politiques de rôle, de fournisseurs SAML, de certificats de serveur, de rôles liés à des services, d'utilisateurs (et d'ajout d'utilisateurs à des groupes), de politiques utilisateur et de dispositifs MFA virtuels. AWS CloudFormation Pour plus d'informations, notamment des exemples de modèles JSON et YAML pour les ressources IAM, consultez la référence au type de AWS Identity and Access Management ressource dans le guide de l'AWS CloudFormation utilisateur.

Vous pouvez également créer des modèles qui créent des ressources connexes, telles que des rôles et des politiques gérées.

En savoir plus sur AWS CloudFormation

Pour en savoir plus AWS CloudFormation, consultez les ressources suivantes :

- AWS CloudFormation
- AWS CloudFormation Guide de l'utilisateur
- AWS CloudFormation API Reference
- AWS CloudFormation Guide de l'utilisateur de l'interface de ligne de commande

Utilisation AWS CloudShell pour travailler avec AWS Identity and Access Management

AWS CloudShell est un shell pré-authentifié basé sur un navigateur que vous pouvez lancer directement depuis le. AWS Management Console Vous pouvez exécuter des AWS CLI commandes sur AWS des services (y compris AWS Identity and Access Management) à l'aide de votre shell

préféré (Bash PowerShell ou Z shell). Et vous pouvez le faire sans télécharger ou installer des outils de ligne de commande.

Vous lancez AWS CloudShell à partir de AWS Management Console, et les AWS informations d'identification que vous avez utilisées pour vous connecter à la console sont automatiquement disponibles dans une nouvelle session shell. Cette pré-authentification des AWS CloudShell utilisateurs vous permet d'ignorer la configuration des informations d'identification lorsque vous interagissez avec AWS des services tels que IAM à l'aide de AWS CLI la version 2 (préinstallée sur l'environnement informatique du shell).

Obtention des autorisations IAM pour AWS CloudShell

À l'aide des ressources de gestion des accès fournies par AWS Identity and Access Management, les administrateurs peuvent accorder des autorisations aux utilisateurs IAM afin qu'ils puissent accéder aux fonctionnalités de l'environnement AWS CloudShell et les utiliser.

Le moyen le plus rapide pour un administrateur d'accorder l'accès aux utilisateurs est d'utiliser une politique AWS gérée. Une politique gérée par AWS est une politique autonome qui est créée et gérée par AWS. La politique AWS gérée suivante pour CloudShell peut être attachée aux identités IAM :

- `AWSCloudShellFullAccess`: accorde l'autorisation d'utilisation AWS CloudShell avec un accès complet à toutes les fonctionnalités.

Si vous souhaitez limiter l'étendue des actions qu'un utilisateur IAM peut effectuer AWS CloudShell, vous pouvez créer une politique personnalisée qui utilise la stratégie `AWSCloudShellFullAccess` gérée comme modèle. Pour plus d'informations sur la limitation des actions disponibles pour les utilisateurs dans CloudShell, consultez la section Gestion de l' AWS CloudShell accès et de l'utilisation avec les politiques IAM dans le Guide de l'AWS CloudShell utilisateur.

Interaction avec IAM à l'aide de AWS CloudShell

Après le lancement AWS CloudShell depuis le AWS Management Console, vous pouvez immédiatement commencer à interagir avec IAM à l'aide de l'interface de ligne de commande.

> **Note**
>
> Lorsque vous AWS CLI l'utilisez AWS CloudShell, vous n'avez pas besoin de télécharger ou d'installer de ressources supplémentaires. De plus, comme vous êtes déjà authentifié dans le

shell, vous n'avez pas besoin de configurer les informations d'identification avant d'effectuer des appels.

Créez un groupe IAM et ajoutez-y un utilisateur IAM à l'aide de AWS CloudShell

L'exemple suivant permet CloudShell de créer un groupe IAM, d'ajouter un utilisateur IAM au groupe, puis de vérifier que la commande a réussi.

1. À partir de AWS Management Console, vous pouvez lancer CloudShell en choisissant les options suivantes disponibles dans la barre de navigation :
   - Choisissez l' CloudShell icône.
   - Commencez à taper « cloudshell » dans le champ de recherche, puis choisissez l' CloudShelloption.
2. Pour créer un groupe IAM, entrez la commande suivante dans la ligne de CloudShell commande. Dans cet exemple, nous avons nommé le groupe east_coast :

   ```
   aws iam create-group --group-name east_coast
   ```

   Si l'appel aboutit, la ligne de commande affiche une réponse du service similaire au résultat suivant :

   ```
   {
      "Group": {
          "Path": "/",
          "GroupName": "east_coast",
          "GroupId": "AGPAYBDBW4JBY3EXAMPLE",
          "Arn": "arn:aws:iam::111122223333:group/east_coast",
          "CreateDate": "2023-09-11T21:02:21+00:00"
       }
   }
   ```

3. Pour ajouter un utilisateur au groupe que vous avez créé, utilisez la commande suivante, en spécifiant le nom du groupe et le nom d'utilisateur. Dans cet exemple, nous avons nommé le groupe east_coast et l'utilisateur johndoe :

```
aws iam add-user-to-group --group-name east_coast --user-name johndoe
```

4. Pour vérifier que l'utilisateur fait partie du groupe, utilisez la commande suivante en spécifiant le nom du groupe. Dans cet exemple, nous continuons à utiliser le groupe east_coast :

```
aws iam get-group --group-name east_coast
```

Si l'appel aboutit, la ligne de commande affiche une réponse du service similaire au résultat suivant :

```
      {
        "Users": [
          {
              "Path": "/",
              "UserName": "johndoe",
              "UserId": "AIDAYBDBW4JBXGEXAMPLE",
              "Arn": "arn:aws:iam::552108220995:user/johndoe",
              "CreateDate": "2023-09-11T20:43:14+00:00",
              "PasswordLastUsed": "2023-09-11T20:59:14+00:00"
          }
        ],
        "Group": {
              "Path": "/",
              "GroupName": "east_coast",
              "GroupId": "AGPAYBDBW4JBY3EXAMPLE",
              "Arn": "arn:aws:iam::111122223333:group/east_coast",
              "CreateDate": "2023-09-11T21:02:21+00:00"
           }
       }
```

Utilisation d'IAM avec un SDK AWS

AWS des kits de développement logiciel (SDK) sont disponibles pour de nombreux langages de programmation populaires. Chaque SDK fournit une API, des exemples de code et de la documentation qui facilitent la création d’applications par les développeurs dans leur langage préféré.

Documentation SDK	Exemples de code
AWS SDK for C++	AWS SDK for C++ exemples de code
AWS CLI	AWS CLI exemples de code
AWS SDK for Go	AWS SDK for Go exemples de code
AWS SDK for Java	AWS SDK for Java exemples de code
AWS SDK for JavaScript	AWS SDK for JavaScript exemples de code
Kit AWS SDK pour Kotlin	Kit AWS SDK pour Kotlin exemples de code
AWS SDK for .NET	AWS SDK for .NET exemples de code
AWS SDK for PHP	AWS SDK for PHP exemples de code
AWS Tools for PowerShell	Outils pour des exemples PowerShell de code
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) exemples de code
AWS SDK for Ruby	AWS SDK for Ruby exemples de code
Kit AWS SDK pour Rust	Kit AWS SDK pour Rust exemples de code
AWS SDK pour SAP ABAP	AWS SDK pour SAP ABAP exemples de code
Kit AWS SDK pour Swift	Kit AWS SDK pour Swift exemples de code

Pour des exemples spécifiques à IAM, consultez Exemples de code pour IAM à l'aide AWS de kits de développement logiciel.

> **Exemple de disponibilité**
>
> Vous n’avez pas trouvé ce dont vous avez besoin ? Demandez un exemple de code en utilisant le lien Provide feedback (Fournir un commentaire) en bas de cette page.

Configuration d'IAM

> **Important**
>
> Les meilleures pratiques IAM recommandent de demander aux utilisateurs humains d'utiliser la fédération avec un fournisseur d'identité pour accéder à l'aide d'informations d'identification temporaires plutôt que d' AWS utiliser des utilisateurs IAM dotés d'informations d'identification à long terme.

AWS Identity and Access Management (IAM) vous aide à contrôler en toute sécurité l'accès à Amazon Web Services (AWS) et aux ressources de votre compte. IAM peut également préserver la confidentialité de vos informations d'identification. Vous ne vous inscrivez pas spécifiquement pour utiliser IAM. L'utilisation d'IAM n'induit aucuns frais.

Utilisez IAM pour donner aux identités, telles que les utilisateurs et les rôles, accès aux ressources de votre compte. Par exemple, vous pouvez utiliser IAM avec les utilisateurs existants de votre annuaire d'entreprise que vous gérez en externe AWS ou vous pouvez créer des utilisateurs en AWS utilisant AWS IAM Identity Center. Les identités fédérées endossent des rôles IAM définis pour accéder aux ressources dont elles ont besoin. Pour plus d'informations sur IAM Identity Center, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

> **Note**
>
> L'IAM est intégré à plusieurs AWS produits. Pour connaître la liste des services prenant en charge IAM, veuillez consulter AWS services qui fonctionnent avec IAM.

Rubriques

- Inscrivez-vous pour un Compte AWS
- Création d'un utilisateur doté d'un accès administratif
- Préparation des autorisations de moindre privilège
- Méthodes de gestion d'IAM
- Votre Compte AWS identifiant et son alias

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez https://portal.aws.amazon.com/billing/signup.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d'inscription, vous recevrez un appel téléphonique et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

   Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l'utilisateur racine a accès à l'ensemble des Services AWS et des ressources de ce compte. La meilleure pratique en matière de sécurité consiste à attribuer un accès administratif à un utilisateur et à n'utiliser que l'utilisateur root pour effectuer les tâches nécessitant un accès utilisateur root.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. Vous pouvez afficher l'activité en cours de votre compte et gérer votre compte à tout moment en accédant à https://aws.amazon.com/ et en choisissant Mon compte.

Création d'un utilisateur doté d'un accès administratif

Une fois que vous vous êtes inscrit à un utilisateur administratif Compte AWS, que vous Utilisateur racine d'un compte AWS l'avez sécurisé AWS IAM Identity Center, que vous l'avez activé et que vous en avez créé un, afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l'aide pour vous connecter en utilisant l'utilisateur racine, consultez Connexion en tant qu'utilisateur racine dans le Guide de l'utilisateur Connexion à AWS .
2. Activez l'authentification multifactorielle (MFA) pour votre utilisateur racine.

Pour obtenir des instructions, consultez la section Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d'un utilisateur doté d'un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d' AWS IAM Identity Center dans le Guide de l'utilisateur AWS IAM Identity Center .
2. Dans IAM Identity Center, accordez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connectez-vous en tant qu'utilisateur disposant d'un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l'URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l'utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribuer l'accès à des utilisateurs supplémentaires

1. Dans IAM Identity Center, créez un ensemble d'autorisations conforme aux meilleures pratiques en matière d'application des autorisations du moindre privilège.

   Pour obtenir des instructions, voir Création d'un ensemble d'autorisations dans le guide de AWS IAM Identity Center l'utilisateur.
2. Affectez des utilisateurs à un groupe, puis attribuez un accès d'authentification unique au groupe.

   Pour obtenir des instructions, consultez la section Ajouter des groupes dans le guide de AWS IAM Identity Center l'utilisateur.

Préparation des autorisations de moindre privilège

L'utilisation d'autorisations de moindre privilège est une recommandation de bonne pratique IAM. Le concept des autorisations de moindre privilège consiste à accorder aux utilisateurs les autorisations nécessaires à l'exécution d'une tâche et aucune autorisation supplémentaire. Lors des préparatifs, réfléchissez à la manière dont vous allez prendre en charge les autorisations de moindre privilège. L'utilisateur root et l'utilisateur administrateur disposent tous deux d'autorisations puissantes qui ne sont pas nécessaires pour les tâches quotidiennes. Pendant que vous découvrez AWS et testez différents services, nous vous recommandons de créer au moins un utilisateur supplémentaire dans IAM Identity Center avec des autorisations moins importantes que vous pourrez utiliser dans différents scénarios. Vous pouvez utiliser les politiques IAM pour définir les actions qui peuvent être entreprises sur des ressources données dans des conditions spécifiques, puis vous connecter à ces ressources avec le compte doté de privilèges moindres.

Si vous utilisez IAM Identity Center, pensez à utiliser des jeux d'autorisations IAM Identity Center pour commencer. Pour en savoir plus, veuillez consulter la rubrique Create a permission set dans le Guide de l'utilisateur IAM Identity Center.

Si vous n'utilisez pas IAM Identity Center, utilisez les rôles IAM pour définir les autorisations des différentes entités IAM. Pour en savoir plus, veuillez consulter la section Création de rôles IAM.

Les rôles IAM et les ensembles d'autorisations IAM Identity Center peuvent utiliser des politiques AWS gérées basées sur les fonctions de travail. Pour plus de détails sur les autorisations accordées par ces politiques, consultez AWS politiques gérées pour les fonctions professionnelles.

> **Important**
>
> N'oubliez pas que les politiques AWS gérées peuvent ne pas accorder d'autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles sont accessibles à tous AWS les clients. Au terme de la configuration, nous vous recommandons d'utiliser IAM Access Analyzer pour générer des stratégies de moindre privilège basées sur l'activité d'accès consignée dans AWS CloudTrail. Pour plus d'informations sur la génération de politiques, consultez IAM Access Analyzer policy generation.

Méthodes de gestion d'IAM

Vous pouvez gérer IAM à l'aide de la AWS console, de l'interface de AWS ligne de commande ou des interfaces d'application (API) des SDK associés. Lors de la configuration, réfléchissez aux méthodes que vous souhaitez prendre en charge et à comment vous envisagez de prendre en charge les différents utilisateurs.

Rubriques

- AWS Console
- AWS Interface de ligne de commande (CLI) et kits de développement logiciel (SDK)

AWS Console

La console AWS de gestion est une application Web qui comprend et fait référence à une vaste collection de consoles de service pour la gestion AWS des ressources. Lors de votre première connexion, vous accédez à la page d'accueil de la console. La page d'accueil permet d'accéder à chaque console de service et propose un emplacement unique pour accéder aux informations nécessaires à l'exécution des tâches AWS connexes. Les services et applications mis à votre disposition une fois connecté à la console dépendent AWS des ressources auxquelles vous êtes autorisé à accéder. Vous pouvez obtenir des autorisations d'accès aux ressources soit en endossant un rôle, soit en étant membre d'un groupe auquel des autorisations ont été accordées, soit en obtenant une autorisation explicite. Pour un AWS compte autonome, l'utilisateur root ou l'administrateur IAM configure l'accès aux ressources. En AWS Organizations effet, le compte de gestion ou l'administrateur délégué configure l'accès aux ressources.

Si vous prévoyez que des personnes utilisent la console de AWS gestion pour gérer les AWS ressources, nous vous recommandons de configurer les utilisateurs avec des informations d'identification temporaires afin de garantir la sécurité. Les utilisateurs IAM qui ont endossé un rôle, les utilisateurs fédérés et les utilisateurs d'IAM Identity Center disposent d'informations d'identification temporaires, tandis que l'utilisateur IAM et l'utilisateur root disposent d'informations d'identification à long terme. Les informations d'identification de l'utilisateur root fournissent un accès complet à l' Compte AWS, tandis que les autres utilisateurs disposent d'informations d'identification leur permettant d'accéder aux ressources qui leur sont accordées par les politiques IAM.

L'expérience de connexion est différente selon les types d' AWS Management Console utilisateurs.

- Les utilisateurs IAM et l'utilisateur root se connectent à partir de l'URL de AWS connexion principale (https://signin.aws.amazon.com). Une fois connectés, ils ont accès aux ressources du compte pour lequel ils ont été autorisés.

  Pour vous connecter en tant qu'utilisateur root, vous devez avoir l'adresse e-mail et le mot de passe de l'utilisateur root.

  Pour vous connecter en tant qu'utilisateur IAM, vous devez disposer du Compte AWS numéro ou de l'alias, du nom d'utilisateur IAM et du mot de passe de l'utilisateur IAM.

  Nous vous recommandons de limiter les utilisateurs IAM de votre compte aux situations spécifiques nécessitant des informations d'identification à long terme, par exemple pour un accès d'urgence, et de n'utiliser l'utilisateur root que pour les tâches nécessitant les informations d'identification de l'utilisateur root.

  Pour plus de commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser le nom d'utilisateur IAM et les informations de compte. La prochaine fois que l'utilisateur accède à une page du AWS Management Console, la console utilise le cookie pour le rediriger vers la page de connexion au compte.

  Déconnectez-vous de la console à la fin de votre session pour empêcher la réutilisation de votre connexion précédente.
- Les utilisateurs d'IAM Identity Center se connectent à l'aide d'un portail d' AWS accès spécifique propre à leur organisation. Une fois connectés, ils peuvent choisir le compte ou l'application auxquels ils veulent accéder. S'ils choisissent d'accéder à un compte, ils choisissent le jeu d'autorisations qu'ils souhaitent utiliser pour la session de gestion.
- Les utilisateurs fédérés gérés par un fournisseur d'identité externe lié à un Compte AWS se connectent à l'aide d'un portail d'accès d'entreprise personnalisé. Les AWS ressources disponibles pour les utilisateurs fédérés dépendent des politiques sélectionnées par leur organisation.

> **Note**
>
> Pour fournir un niveau de sécurité supplémentaire, l'utilisateur root, les utilisateurs IAM et les utilisateurs d'IAM Identity Center peuvent faire vérifier l'authentification multifactorielle (MFA) AWS avant d'accorder l'accès aux ressources. AWS Lorsque l'authentification MFA est activée, vous devez également avoir accès au périphérique MFA pour vous connecter.

Pour en savoir plus sur la façon dont les différents utilisateurs se connectent à la console de gestion, voir Connexion à la console de AWS gestion dans le Guide de l'utilisateur de AWS connexion.

AWS Interface de ligne de commande (CLI) et kits de développement logiciel (SDK)

Les utilisateurs IAM Identity Center et les utilisateurs IAM utilisent différentes méthodes pour authentifier leurs informations d'identification lorsqu'ils s'authentifient par le biais de la CLI ou des interfaces d'application (API) des kits SDK associés.

Les informations d'identification et les paramètres de configuration se trouvent à plusieurs endroits, tels que les variables d'environnement système ou utilisateur, les fichiers de AWS configuration locaux, ou sont explicitement déclarés sur la ligne de commande en tant que paramètre. Certains emplacements l'emportent sur d'autres.

IAM Identity Center et IAM fournissent des clés d'accès qui peuvent être utilisées avec la CLI ou le kit SDK. Les clés d'accès IAM Identity Center sont des identifiants temporaires qui peuvent être actualisés automatiquement et leur utilisation est recommandée par rapport aux clés d'accès à long terme associées aux utilisateurs IAM.

Pour gérer votre Compte AWS utilisation de la CLI ou du SDK, vous pouvez utiliser AWS CloudShell votre navigateur. Si vous exécutez CloudShell des commandes CLI ou SDK, vous devez d'abord vous connecter à la console. Les autorisations d'accès aux AWS ressources sont basées sur les informations d'identification que vous avez utilisées pour vous connecter à la console. Selon votre expérience, vous trouverez peut-être que la CLI est une méthode plus efficace pour gérer votre Compte AWS.

Pour le développement d'applications, vous pouvez télécharger la CLI ou le kit SDK sur votre ordinateur et vous connecter à partir de l'invite de commande ou d'une fenêtre Docker. Dans ce scénario, vous configurez les informations d'identification relatives à l'authentification et à l'accès dans le cadre du script CLI ou de l'application SDK. Vous pouvez configurer l'accès programmatique aux ressources de différentes manières, en fonction de l'environnement et de l'accès dont vous disposez.

- Les options recommandées pour authentifier le code local avec le AWS service sont IAM Identity Center et IAM Roles Anywhere.
- Les options recommandées pour authentifier le code exécuté dans un AWS environnement sont d'utiliser des rôles IAM ou des informations d'identification IAM Identity Center.

Si vous utilisez IAM Identity Center, vous pouvez obtenir des informations d'identification à court terme depuis la page d'accueil du portail d'accès AWS où vous choisissez votre jeu d'autorisations. Ces informations d'identification ont une durée définie et ne sont pas actualisées automatiquement. Si vous souhaitez utiliser ces informations d'identification, après vous être connecté au AWS portail, choisissez le, Compte AWS puis choisissez l'ensemble d'autorisations. Sélectionnez Accès par ligne de commande ou par programmation pour afficher les options que vous pouvez utiliser pour accéder aux AWS ressources par programmation ou à partir de la CLI. Pour plus d'informations sur ces méthodes, veuillez consulter la rubrique Getting and refreshing temporary credentials dans le Guide de l'utilisateur IAM Identity Center. Ces informations d'identification sont souvent utilisées lors du développement d'applications pour tester rapidement le code.

Nous vous recommandons d'utiliser les informations d'identification IAM Identity Center qui sont automatiquement actualisées lors de l'automatisation de l'accès à vos AWS ressources. Si vous avez configuré des utilisateurs et des jeux d'autorisations dans IAM Identity Center, utilisez la commande `aws configure sso` pour faire appel à un assistant de ligne de commande qui vous aidera à identifier les informations d'identification disponibles et à les stocker dans un profil. Pour plus d'informations sur la configuration de votre profil, veuillez consulter la rubrique Configure your profile with the `aws configure sso` wizard dans le Guide de l'utilisateur de l'interface de ligne de commande AWS pour la version 2.

> **Note**
>
> De nombreux exemples d'applications utilisent des clés d'accès à long terme associées à des utilisateurs IAM ou à un utilisateur root. N'utilisez les informations d'identification à long terme qu'au sein d'un environnement de test (sandbox), dans le cadre d'un exercice d'apprentissage. Passez en revue les alternatives aux clés d'accès à long terme et planifiez la modification de votre code pour utiliser des informations d'identification alternatives, telles que les informations d'identification IAM Identity Center ou les rôles IAM, dès que possible. Après avoir modifié votre code, supprimez les clés d'accès.

Pour en savoir plus sur la configuration de la CLI, voir Installer ou mettre à jour la dernière version de la AWS CLI dans le Guide de l'utilisateur de l'interface de ligne de AWS commande pour la version 2 et Authentification et informations d'identification d'accès dans le Guide de l'utilisateur de l'interface de ligne de AWS commande

Pour en savoir plus sur la configuration du kit SDK, veuillez consulter les rubriques Authentification IAM Identity Center et IAM Roles Anywhere dans le Guide de référence des kits SDK et outils AWS .

Votre Compte AWS identifiant et son alias

Les utilisateurs IAM du compte se connectent à l'aide d'une URL Web qui comprend soit l'alias du compte, soit un ID du compte. Si vous n'avez pas l'URL, la page de AWS connexion nécessite que vous fournissiez l' Compte AWS alias ou l'identifiant du compte.

Si vous ne connaissez pas l'identifiant ou l'alias de votre compte :

- Vérifiez l'historique de votre navigateur. Si vous vous êtes déjà connecté, il est possible qu'il soit enregistré dans vos sites Web récents.
- Si vous avez configuré la AWS CLI ou un AWS SDK avec les informations d'identification de votre compte, vous pouvez obtenir votre identifiant de compte à partir de vos fichiers de configuration.
- Demandez à votre administrateur local ou au propriétaire du compte, vous AWS ne pouvez pas fournir d'identifiant de compte aux utilisateurs.

> **Tip**
>
> Pour créer un marque-page pour la page de connexion à votre compte dans votre navigateur web, vous devez entrer manuellement l'URL de connexion lors de la création du marque-page. N'utilisez pas la fonctionnalité « Ajouter cette page aux favoris » de votre navigateur Web, car elle capture des informations spécifiques à votre session de navigation en cours qui interfèrent avec vos prochaines visites sur la page de connexion.

Rubriques

- Afficher votre Compte AWS identifiant
- À propos des alias de compte
- Création, suppression et affichage d'un alias d' Compte AWS

Afficher votre Compte AWS identifiant

Vous pouvez consulter l'identifiant de votre compte Compte AWS en utilisant les méthodes suivantes.

Affichez votre ID de compte à l'aide de l'outil

L'identifiant du compte est affiché sur le tableau de bord IAM dans la Compte AWS section. Il existe d'autres moyens de consulter l'identifiant de votre compte dans la console en fonction de votre type

d'utilisateur. Si vous avez assumé un rôle, les informations d'identification de sécurité ne sont pas disponibles.

Type utilisateur	Procédure
Utilisateur root	Dans la barre de navigation en haut à droite, choisissez votre nom d'utilisateur, puis sélectionnez Security credentials. Le numéro de compte apparaît sous Identifiants de compte.
Utilisateur IAM	Dans la barre de navigation en haut à droite, choisissez votre nom d'utilisateur, l'identifiant du compte est affiché au-dessus de votre nom d'utilisateur. Choisissez Informations d’identif ication de sécurité. Le numéro de compte s'affiche sous Détails du compte.
Utilisateur fédéré	Dans la barre de navigation en haut à droite, choisissez votre nom d'utilisateur, l'identifiant du compte est affiché au-dessus de votre nom d'utilisateur.
Rôle endossé	Dans la barre de navigation en haut à droite, choisissez l'icône Support, puis sélectionnez Support Center dans la liste. Le numéro de compte (ID) à 12 chiffres avec lequel vous êtes actuellement connecté apparaît dans le panneau de navigation Support Center (Centre de support).

Consultez l'identifiant de votre compte à l'aide du AWS CLI

Utilisez la commande suivante pour afficher votre ID utilisateur, votre ID de compte et votre ARN utilisateur :

- ensembles de lois get-caller-identity

Affichez votre ID de compte à l'aide de l'API

Utilisez l'API suivante pour afficher votre ID utilisateur, votre ID de compte et votre ARN utilisateur :

- GetCallerIdentity

À propos des alias de compte

Si vous souhaitez que l'URL de votre page de connexion contienne le nom de votre entreprise (ou un autre identifiant convivial) au lieu de votre Compte AWS identifiant, vous pouvez créer un alias de compte. Cette section fournit des informations sur Compte AWS les alias et répertorie les opérations d'API que vous utilisez pour créer un alias.

Par défaut, l'URL de votre page de connexion utilise le format suivant.

```
https://Your_Account_ID.signin.aws.amazon.com/console/
```

Si vous créez un Compte AWS alias pour votre Compte AWS identifiant, l'URL de votre page de connexion ressemble à l'exemple suivant.

```
https://Your_Account_Alias.signin.aws.amazon.com/console/
```

Considérations

- Vous ne Compte AWS pouvez avoir qu'un seul alias. Si vous créez un alias pour votre compte AWS , celui-ci remplace l'ancien alias et l'URL contenant l'ancien alias cesse de fonctionner.
- L'alias du compte ne doit contenir que des chiffres, des lettres minuscules et des traits d'union. Pour plus d'informations sur les limitations relatives aux entités de AWS compte, consultezIAM et quotas AWS STS.
- L'alias du compte doit être unique pour tous les produits Amazon Web Services dans une partition de réseau donnée.

  Une partition est un groupe de AWS régions. Chaque compte AWS est étendu à une partition.

  Les partitions prises en charge sont les suivantes :

  - `aws`- AWS Régions
  - `aws-cn` - Régions chinoises

- `aws-us-gov-` AWS GovCloud (US) Régions

Création, suppression et affichage d'un alias d' Compte AWS

Vous pouvez utiliser l' AWS Management Console API IAM ou l'interface de ligne de commande pour créer ou supprimer votre Compte AWS alias.

> **Note**
>
> Les alias de compte ne sont pas des secrets et ils apparaîtront dans l'URL de votre page de connexion publique. N'incluez aucune information sensible dans l'alias de votre compte. L'URL d'origine contenant votre Compte AWS identifiant reste active et peut être utilisée une fois que vous avez créé votre Compte AWS alias.

Créer ou modifier un alias de compte (console)

Vous pouvez créer, modifier et supprimer un alias de compte à partir de la AWS Management Console.

> **Autorisations minimales**
>
> Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :
>
> - `iam:ListAccountAliases`
> - `iam:CreateAccountAlias`

Créer ou modifier un alias de compte (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Dashboard (Tableau de bord).
3. Dans la section AWS Compte, près de Alias du compte, choisissez Créer. Si un alias existe déjà, sélectionnez Edit (Modifier).

4. Dans la boîte de dialogue, saisissez le nom que vous souhaitez utiliser pour l'alias, puis sélectionnez Enregistrer les modifications.

> **Note**
>
> Vous ne pouvez avoir qu'un seul alias associé Compte AWS à votre nom à la fois. Si vous créez un nouvel alias, l'alias précédent est supprimé et l'URL de connexion associée à l'alias précédent s'arrête de fonctionner.

Supprimer un alias de compte (outil)

Vous pouvez supprimer un alias de compte à partir de la AWS Management Console.

> **Autorisations minimales**
>
> Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :
>
> - `iam:ListAccountAliases`
> - `iam:CreateAccountAlias`
> - `iam:DeleteAccountAlias`

Pour supprimer un alias de compte (outil)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Dashboard (Tableau de bord).
3. Dans la section AWS Compte, près de l'Alias du compte, choisissez Supprimer.

> **Note**
>
> La seule URL de connexion à votre compte dépend de votre identifiant de compte. Les tentatives de connexion à l'URL de l'alias ne sont pas redirigées.

Création, suppression et affichage d'alias (AWS CLI)

> **Note**
>
> Pour utiliser les commandes suivantes, vous devez disposer au moins des autorisations IAM suivantes :
>
> - `iam:ListAccountAliases`
> - `iam:CreateAccountAlias`
> - `iam:DeleteAccountAlias`

Pour créer un alias pour l'URL de votre page de AWS Management Console connexion, exécutez la commande suivante :

- `aws iam create-account-alias`

Pour supprimer un alias d' Compte AWS ID, exécutez la commande suivante :

- `aws iam delete-account-alias`

Pour afficher l'alias de votre Compte AWS identifiant, exécutez la commande suivante :

- `aws iam list-account-aliases`

Example Commandes d'alias

Pour afficher l'alias de votre Compte AWS identifiant, exécutez la commande suivante.

```
$ aws iam list-account-aliases
{
    "AccountAliases": [
        "myaccountalias"
    ]
}
```

Pour créer un alias pour votre AWS Management Console connexion, exécutez la commande suivante :

```
$ aws iam create-account-alias \
    --account-alias myaliasname
```

Cette commande ne produit aucune sortie si elle réussit.

Pour supprimer un alias d' Compte AWS ID, exécutez la commande suivante.

```
$ aws iam delete-account-alias \
--account-alias myaliasname
```

Cette commande ne produit aucune sortie si elle réussit.

Création, suppression et affichage d'alias (API AWS)

> **Note**
>
> Pour utiliser les opérations d'API suivantes, vous devez disposer au moins des autorisations IAM suivantes :
>
> - `iam:ListAccountAliases`
> - `iam:CreateAccountAlias`
> - `iam:DeleteAccountAlias`

Pour créer un alias pour l'URL de votre page de AWS Management Console connexion, effectuez l'opération suivante :

- `CreateAccountAlias`

Pour supprimer un alias d' Compte AWS ID, effectuez l'opération suivante :

- `DeleteAccountAlias`

Pour afficher l'alias de votre Compte AWS identifiant, effectuez l'opération suivante :

- `ListAccountAliases`

Mise en route avec IAM

Utilisez ce didacticiel pour démarrer avec AWS Identity and Access Management (IAM). Vous allez apprendre à créer des rôles, des utilisateurs et des politiques à l'aide de la AWS Management Console.

AWS Identity and Access Management est une fonctionnalité de votre Compte AWS offre sans frais supplémentaires. L'utilisation d'autres AWS produits par vos utilisateurs IAM vous sera facturée uniquement. Pour plus d'informations sur les prix des autres AWS produits, consultez la page de tarification d'Amazon Web Services.

> **Note**
>
> Cet ensemble de documentation traite principalement du service IAM. Pour en savoir plus sur la prise en main AWS et l'utilisation de plusieurs services pour résoudre un problème tel que la création et le lancement de votre premier projet, consultez le Centre de ressources pour la mise en route.

Table des matières

- Prérequis
- Création de votre premier utilisateur IAM
- Création de votre premier rôle
- Création de votre première politique IAM
- Accès par programmation

Prérequis

Avant de commencer, assurez-vous d'avoir terminé les étapes de Configuration d'IAM. Ce didacticiel utilise le compte administrateur que vous avez créé au cours de cette procédure.

Création de votre premier utilisateur IAM

Un utilisateur IAM est une identité au sein de vous Compte AWS qui possède des autorisations spécifiques pour une seule personne ou application. Les utilisateurs peuvent être organisés en groupes qui partagent les mêmes autorisations.

> **Note**
>
> Une bonne pratique en matière de sécurité consiste à fournir un accès à vos ressources par le biais de la fédération d'identité plutôt que de créer des utilisateurs IAM. Pour en savoir plus sur les situations spécifiques dans lesquelles un utilisateur IAM est nécessaire, veuillez consulter Quand créer un utilisateur IAM (au lieu d'un rôle).

Pour vous permettre de vous familiariser avec le processus de création d'un utilisateur IAM, ce didacticiel explique comment créer un utilisateur et un groupe IAM dans le cadre d'un accès d'urgence.

Pour créer votre premier utilisateur IAM

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d'accueil de la console, sélectionnez le service IAM.
3. Dans le volet de navigation, sélectionnez Utilisateurs, puis Ajouter des utilisateurs.

   > **Note**
   >
   > Si IAM Identity Center est activé, un rappel indiquant qu'il est préférable de gérer l'accès des utilisateurs dans IAM Identity Center s' AWS Management Console affiche. Dans le cadre de ce didacticiel, l'utilisateur IAM que vous créez est spécifiquement destiné à n'être utilisé que lorsque les informations d'identification de votre utilisateur ne sont pas disponibles dans IAM Identity Center.

4. Dans User name (Nom d'utilisateur), saisissez **`EmergencyAccess`**. Les noms ne peuvent pas contenir d'espaces.
5. Cochez la case à côté de Fournir un accès utilisateur au AWS Management Console— facultatif, puis choisissez Je souhaite créer un utilisateur IAM.
6. Sous Mot de passe de la console, sélectionnez Mot de passe généré automatiquement.
7. Décochez la case située en regard de l'option L'utilisateur doit créer un nouveau mot de passe à la prochaine connexion (recommandé). Comme cet utilisateur IAM est destiné à un accès d'urgence, un administrateur de confiance conserve le mot de passe et ne le fournit qu'en cas de besoin.

8. Sur la page Définir les autorisations, sous Options d'autorisations, sélectionnez Ajouter un utilisateur au groupe. Ensuite, sous Groupes d'utilisateurs, sélectionnez Créer un groupe.
9. Sur la page Créer un groupe d'utilisateurs, saisissez **`EmergencyAccessGroup`** dans le champ Nom du groupe d'utilisateurs. Ensuite, sous Politiques d'autorisations, sélectionnez AdministratorAccess.
10. Sélectionnez Créer un groupe d'utilisateurs pour revenir à la page Définir les autorisations.
11. Sous Groupes d'utilisateurs, sélectionnez le nom du groupe **`EmergencyAccessGroup`** précédemment créé.
12. Sélectionnez Suivant pour accéder à la page Vérifier et créer.
13. Sur la page Vérifier et créer, examinez la liste des appartenances aux groupes à attribuer au nouvel utilisateur. Une fois que vous êtes prêt à continuer, sélectionnez Créer l'utilisateur.
14. Sur la page Récupérer le mot de passe, sélectionnez Télécharger le fichier .csv pour enregistrer le fichier .csv contenant les informations d'identification de l'utilisateur (URL de connexion, nom d'utilisateur et mot de passe).
15. Enregistrez ce fichier pour l'utiliser si vous devez vous connecter à IAM et que vous n'avez pas accès à votre fournisseur d'identité fédérée.

Le nouvel utilisateur IAM apparaît dans la liste Utilisateurs. Cliquez sur le lien Nom d'utilisateur pour afficher les détails de l'utilisateur. Sous Résumé, copiez l'ARN de l'utilisateur dans le presse-papiers. Collez l'ARN dans un document texte afin de pouvoir l'utiliser au cours de la procédure suivante.

Création de votre premier rôle

Les rôles IAM constituent un moyen sécurisé d'accorder des autorisations à des entités de confiance. Un rôle IAM présente des similitudes avec un utilisateur IAM. Les rôles et les utilisateurs sont des principaux associés à des politiques d'autorisations qui déterminent ce que l'identité peut ou ne peut pas faire dans AWS. En revanche, au lieu d'être associé de manière unique à une personne, un rôle est conçu pour être endossé par tout utilisateur qui en a besoin. En outre, un rôle ne dispose pas d’informations d’identification standard à long terme comme un mot de passe ou des clés d’accès associées. Au lieu de cela, lorsque vous adoptez un rôle, il vous fournit des informations d’identification de sécurité temporaires pour votre session de rôle. L'utilisation de rôles vous permet de suivre les meilleures pratiques en matière d'IAM. Vous pouvez utiliser un rôle pour :

- Activez les identités du personnel et permettez aux applications activées par Identity Center d'accéder à AWS Management Console l'utilisateur AWS IAM Identity Center.

- Déléguez l'autorisation à un AWS service d'effectuer des actions en votre nom.
- Activer le code d'application s'exécutant sur une instance Amazon EC2 pour accéder à des ressources AWS ou les modifier.
- Accordez l'accès à un autre Compte AWS.

> **Note**
>
> Vous pouvez utiliser AWS Identity and Access Management Roles Anywhere pour donner accès aux identités des machines. L'utilisation d'IAM Roles Anywhere signifie que vous n'avez pas besoin de gérer les informations d'identification à long terme pour les charges de travail exécutées en dehors de. AWS Pour plus d'informations, consultez Présentation de Rôles Anywhere AWS Identity and Access Management dans le Guide de l'utilisateur de Rôles Anywhere AWS Identity and Access Management .

IAM Identity Center et d'autres AWS services créent automatiquement des rôles pour leurs services. Si vous utilisez des utilisateurs IAM, nous vous recommandons de leur créer des rôles qu'ils pourront endosser lorsqu'ils se connecteront. Ils disposeront ainsi d'autorisations temporaires valables pour la durée de la session plutôt que d'autorisations à long terme.

L' AWS Management Console assistant qui vous guide à travers les étapes de création d'un rôle affiche des étapes légèrement différentes selon que vous créez un rôle pour un utilisateur IAM, un AWS service ou un utilisateur fédéré. L'accès régulier au Comptes AWS sein d'une organisation doit être fourni à l'aide d'un accès fédéré. Si vous créez des utilisateurs IAM à des fins spécifiques, telles qu'un accès d'urgence ou un accès programmatique, contentez-vous d'accorder à ces utilisateurs IAM l'autorisation d'endosser un rôle et placez-les dans des groupes propres à un rôle.

Dans cette procédure, vous créez un rôle qui fournit un SupportUser accès à l'utilisateur EmergencyAccess IAM. Avant d'entamer cette procédure, copiez l'ARN de l'utilisateur IAM dans le presse-papiers.

Pour créer un rôle pour un utilisateur IAM

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d'accueil de la console, sélectionnez le service IAM.

3. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
4. Choisissez le type du rôle de l'Compte AWS.
5. Dans Sélectionner une entité approuvée, accédez au champ Type d'entité approuvée et choisissez Politique d'approbation personnalisée.
6. Dans la section Politique de confiance personnalisée, examinez la politique d'approbation de base. Il s'agit de la politique que nous utiliserons pour ce rôle. Utilisez l'éditeur Modifier l'instruction pour mettre à jour la politique d'approbation :

   1. Dans Ajouter des actions pour STS, sélectionnez Endosser un rôle.
   2. En regard de l'option Ajouter un principal, sélectionnez Ajouter. La fenêtre Ajouter un principal s'ouvre.

      Sous Type de principal, sélectionnez Utilisateurs IAM.

      Sous ARN, collez l'ARN de l'utilisateur IAM que vous avez copié dans le presse-papiers.

      Sélectionnez Ajouter le principal.
   3. Vérifiez que la ligne `Principal` de la politique d'approbation contient maintenant l'ARN que vous avez spécifié :

      ```
      "Principal": { "AWS": "arn:aws:iam::123456789012:user/username" }
      ```
7. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant.
8. Dans Ajouter des autorisations, cochez la case située en regard de la politique d'approbation à appliquer. Pour ce didacticiel, nous allons sélectionner la politique de SupportUserconfiance. Vous pouvez ensuite utiliser ce rôle pour résoudre les problèmes liés à Compte AWS et ouvrir des dossiers de support auprès AWS de. Nous n'allons pas définir de limite d'autorisations pour le moment.
9. Choisissez Suivant.
10. Dans Nommer, vérifier et créer, définissez les paramètres suivants :

    - Dans Nom du rôle, entrez un nom identifiant ce rôle, tel que SupportUserRole.
    - Dans Description, expliquez à quoi doit servir le rôle.

Comme d'autres AWS ressources peuvent faire référence au rôle, vous ne pouvez pas modifier le nom du rôle une fois celui-ci créé.

11. Sélectionnez Créer le rôle.

    Une fois le rôle créé, transmettez les informations relatives à celui-ci aux personnes qui en ont besoin. Vous pouvez transmettre les informations relatives au rôle sous les formes suivantes :

    - Lien du rôle : envoyez aux utilisateurs un lien qui les dirige vers la page Switch Role (Changer de rôle) dans laquelle tous les détails sont déjà remplis.
    - ID ou alias du compte : donnez à chaque utilisateur le nom du rôle ainsi que l'ID ou l'alias du compte. L'utilisateur accède ensuite à la page Switch Role (Changer de rôle) et ajoute les détails manuellement.
    - Enregistrer les informations du lien du rôle ainsi que les informations EmergencyAccess d'identification de l'utilisateur.

    Pour plus de détails, consultez Fourniture d'informations à l'utilisateur.

Création de votre première politique IAM

Les politiques IAM sont attachées à des identités IAM (utilisateurs, groupes d'utilisateurs ou rôles) ou à des ressources AWS . Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit leurs autorisations.

Pour créer votre première politique IAM

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d'accueil de la console, sélectionnez le service IAM.
3. Dans le panneau de navigation, choisissez Politiques.

   Si vous sélectionnez Politiques pour la première fois, la page Bienvenue dans les politiques gérées s’affiche. Sélectionnez Get started (Mise en route).
4. Sélectionnez Créer une politique.
5. Sur la page Créer une politique, sélectionnez Actions, puis choisissez Importer une politique.

6. Dans la fenêtre Importer une politique, dans le champ Rechercher des politiques, saisissez **`power`** afin de réduire la liste des politiques. Sélectionnez la PowerUserAccesspolitique.
7. Sélectionnez Importer une politique. La politique apparaît dans l'onglet JSON.
8. Choisissez Suivant.
9. Sur la page Vérifier et créer, pour le Nom de la politique, saisissez **`PowerUserExamplePolicy`**. Pour Description, saisissez **`Allows full access to all services except those for user management`**. Sélectionnez ensuite Créer une politique pour enregistrer la politique.

Vous pouvez attacher cette politique à un rôle afin de fournir aux utilisateurs qui endossent ce rôle les autorisations associées à cette politique. La PowerUserAccesspolitique est couramment utilisée pour fournir un accès aux développeurs.

Accès par programmation

Les utilisateurs ont besoin d'un accès programmatique s'ils souhaitent interagir avec AWS l'extérieur du AWS Management Console. La manière d'accorder un accès programmatique dépend du type d'utilisateur accédant AWS :

- Si vous gérez les identités dans IAM Identity Center, les AWS API nécessitent un profil et un profil ou une variable d'environnement. AWS Command Line Interface
- Si vous avez des utilisateurs IAM, les AWS API et les clés AWS Command Line Interface d'accès requises. Lorsque cela est possible, créez des informations d’identification temporaires composées d’un ID de clé d’accès, d’une clé d’accès secrète et d’un jeton de sécurité qui indique la date d’expiration des informations d’identification.

Pour accorder aux utilisateurs un accès programmatique, choisissez l’une des options suivantes.

Quel utilisateur a besoin d’un accès programmatique ?	Pour	Par
Identité de la main-d’œuvre (Utilisateurs gérés dans IAM Identity Center)	Utilisez des informations d'identification à court terme pour signer les demandes programmatiques adressées	Suivez les instructions de l'interface que vous souhaitez utiliser :

Quel utilisateur a besoin d'un accès programmatique ?	Pour	Par
	aux AWS API AWS CLI or (directement ou à l'aide des AWS SDK).	• Pour ce faire AWS CLI, suivez les instructions de la section Obtenir les informations d'identification du rôle IAM pour l'accès à la CLI dans le guide de AWS IAM Identity Center l'utilisateur. • Pour les AWS API, suivez les instructions figurant dans les informations d'identification SSO du Guide de référence AWS des SDK et des outils.
IAM	Utilisez des informations d'identification à court terme pour signer les demandes programmatiques adressées aux AWS API AWS CLI or (directement ou à l'aide des AWS SDK).	Suivez les instructions de la section Utilisation d'informations d'identification temporaires avec AWS les ressources.
IAM	Utilisez des informations d'identification à long terme pour signer les demandes programmatiques adressées aux AWS API AWS CLI or (directement ou à l'aide des AWS SDK). (Non recommandé)	Suivez les instructions de la section Gestion des clés d'accès pour les utilisateurs IAM.

Bonnes pratiques de sécurité et cas d'utilisation dans AWS Identity and Access Management

AWS Identity and Access Management (IAM) fournit un certain nombre de fonctionnalités de sécurité à prendre en compte lors de l'élaboration et de la mise en œuvre de vos propres politiques de sécurité. Les bonnes pratiques suivantes doivent être considérées comme des instructions générales et ne représentent pas une solution de sécurité complète. Étant donné que ces bonnes pratiques peuvent ne pas être appropriées ou suffisantes pour votre environnement, considérez-les comme des remarques utiles plutôt que comme des recommandations.

Pour tirer profit au maximum d'IAM, prenez le temps de découvrir les bonnes pratiques recommandées. Pour cela, découvrez comment IAM est utilisé dans des scénarios concrets afin d'utiliser les autres services AWS .

Rubriques

- Bonnes pratiques de sécurité dans IAM
- Bonnes pratiques d'utilisateur root pour votre Compte AWS
- Cas d'utilisation métier pour IAM

Bonnes pratiques de sécurité dans IAM

Follow us on Twitter

Les AWS Identity and Access Management meilleures pratiques ont été mises à jour le 14 juillet 2022.

Pour sécuriser vos AWS ressources, suivez ces bonnes pratiques pour AWS Identity and Access Management (IAM).

Rubriques

- Obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires
- Exiger que les charges de travail utilisent des informations d'identification temporaires associées à des rôles IAM pour y accéder AWS

- Authentification multifactorielle (MFA) nécessaire.
- Mettre à jour les clés d'accès lorsque cela est nécessaire pour les cas d'utilisation nécessitant des informations d'identification à long terme
- Suivez les meilleures pratiques pour protéger vos informations d'identification de l'utilisateur root
- Accorder les autorisations de moindre privilège
- Commencez avec les politiques AWS gérées et passez aux autorisations du moindre privilège
- Utiliser IAM Access Analyzer pour générer des politiques de moindre privilège en fonction de l'activité d'accès
- Vérifiez et supprimez régulièrement les utilisateurs, les rôles, les autorisations, les politiques et les informations d'identification non utilisés
- Utiliser des conditions dans les stratégies IAM pour restreindre davantage l'accès
- Vérifiez l'accès public et intercompte aux ressources avec IAM Access Analyzer.
- Utilisez IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles
- Établir des garde-fous d'autorisations sur plusieurs comptes
- Utiliser les limites des autorisations pour déléguer la gestion des autorisations au sein d'un compte.

Obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires

Les utilisateurs humains, également connus sous le nom identités humaines, sont les personnes, les administrateurs, les développeurs, les opérateurs et les consommateurs de vos applications. Ils doivent disposer d'une identité pour accéder à vos AWS environnements et applications. Les utilisateurs humains membres de votre organisation sont également appelés Identités de personnel. Les utilisateurs humains peuvent également être des utilisateurs externes avec lesquels vous collaborez et qui interagissent avec vos AWS ressources. Ils peuvent le faire via un navigateur Web, une application client, une application mobile ou des outils de ligne de commande interactifs.

Exigez de vos utilisateurs humains qu'ils utilisent des informations d'identification temporaires lors de l'accès AWS. Vous pouvez utiliser un fournisseur d'identité auquel vos utilisateurs humains fourniront un accès fédéré Comptes AWS en assumant des rôles fournissant des informations d'identification temporaires. Pour une gestion centralisée des accès, nous vous recommandons d'utiliser AWS

IAM Identity Center (IAM Identity Center) pour gérer l'accès à vos comptes et les autorisations au sein de ceux-ci. Vous pouvez gérer vos identités d'utilisateur avec IAM Identity Center ou gérer les autorisations d'accès pour les identités des utilisateurs dans IAM Identity Center à partir d'un fournisseur d'identité externe. Pour plus d'informations, consultez Présentation de AWS IAM Identity Center dans le Guide de l'utilisateur AWS IAM Identity Center .

Pour plus d'informations sur les rôles , consultez Termes et concepts relatifs aux rôles.

Exiger que les charges de travail utilisent des informations d'identification temporaires associées à des rôles IAM pour y accéder AWS

Une charge de travail est un ensemble de ressources et de code qui fournit une valeur business, par exemple une application destinée au client ou un processus de backend. Votre charge de travail peut comporter des applications, des outils opérationnels et des composants qui nécessitent une identité pour envoyer des demandes à Services AWS, telles que les demandes de lecture de données. Ces identités incluent les machines exécutées dans vos AWS environnements, telles que les instances ou AWS Lambda les fonctions Amazon EC2.

Vous pouvez également gérer les Identités machine pour les parties externes qui ont besoin d'un accès. Pour donner accès aux identités des machines, vous pouvez utiliser des rôles IAM. Les rôles IAM disposent d'autorisations spécifiques et fournissent un moyen d'accès AWS en s'appuyant sur des informations d'identification de sécurité temporaires associées à une session de rôle. En outre, il se peut AWS que des machines extérieures aient besoin d'accéder à vos AWS environnements. Pour les machines qui s'exécutent à l'extérieur, AWS vous pouvez utiliser AWS Identity and Access Management Roles Anywhere. Pour plus d'informations sur les rôles , consultez Rôles IAM. Pour plus de détails sur l'utilisation des rôles pour déléguer l'accès entre Comptes AWS eux, consultezDidacticiel IAM : déléguer l'accès entre des comptes AWS à l'aide des rôles IAM.

Authentification multifactorielle (MFA) nécessaire.

Nous recommandons d'utiliser les rôles IAM pour les utilisateurs humains et les charges de travail qui accèdent à vos ressources AWS afin qu'ils utilisent des informations d'identification temporaires. Toutefois, pour les scénarios dans lesquels vous avez besoin d'un utilisateur IAM ou root dans votre compte, exigez une MFA pour plus de sécurité. Avec MFA, les utilisateurs disposent d'un périphérique qui génère une réponse à une stimulation d'authentification. Les informations d'identification de l'utilisateur et la réponse générée par le périphérique sont requis pour mener à bien le processus du connexion. Pour plus d'informations, consultez Utilisation de l'authentification multifactorielle (MFA) dans AWS.

Si vous utilisez IAM Identity Center pour la gestion centralisée des accès pour les utilisateurs humains, vous pouvez utiliser les fonctionnalités MFA d'IAM Identity Center lorsque votre source d'identité est configurée avec le magasin d'identités IAM Identity Center, Managed AWS Microsoft AD ou AD Connector. Pour de plus d'informations sur la MFA dans IAM Identity Center, consultez Authentification multi-facteur dans le Guide de l'utilisateur AWS IAM Identity Center .

Mettre à jour les clés d'accès lorsque cela est nécessaire pour les cas d'utilisation nécessitant des informations d'identification à long terme

Dans la mesure du possible, nous vous recommandons de vous appuyer sur des informations d'identification temporaires plutôt que de créer des informations d'identification à long terme tels que les clés d'accès. Toutefois, pour les scénarios dans lesquels vous avez besoin d'utilisateurs IAM disposant d'un accès par programmation et d'informations d'identification à long terme, nous vous recommandons de mettre à jour les clés d'accès lorsque cela est nécessaire, par exemple lorsqu'un employé quitte votre entreprise. Nous vous recommandons d'utiliser les dernières informations consultées relatives à un accès IAM pour mettre à jour et retirer les clés d'accès en toute sécurité. Pour plus d’informations, consultez Mise à jour des clés d'accès.

Certains cas d'utilisation spécifiques nécessitent des informations d'identification à long terme avec les utilisateurs IAM dans AWS. Voici certaines des fonctionnalités les plus utilisées :

- Charges de travail qui ne peuvent pas utiliser de rôles IAM — Vous pouvez exécuter une charge de travail à partir d'un emplacement qui a besoin d'accéder à AWS. Dans certains cas, vous ne pouvez pas utiliser les rôles IAM pour fournir des informations d'identification temporaires, par exemple pour les WordPress plug-ins. Dans ces situations, utilisez les clés d'accès à long terme de l'utilisateur IAM pour cette charge de travail afin de vous authentifier auprès de AWS.
- AWS Clients tiers : si vous utilisez des outils qui ne prennent pas en charge l'accès avec IAM Identity Center, tels que des AWS clients tiers ou des fournisseurs qui ne sont pas hébergés sur le site AWS, utilisez les clés d'accès à long terme des utilisateurs IAM.
- AWS CodeCommit accès — Si vous avez l'habitude de CodeCommit stocker votre code, vous pouvez utiliser un utilisateur IAM doté de clés SSH ou d'informations d'identification spécifiques au service pour vous authentifier CodeCommit auprès de vos référentiels. Nous vous recommandons de procéder ainsi en plus de vous servir d'un utilisateur dans IAM Identity Center pour une authentification ordinaire. Les utilisateurs d'IAM Identity Center sont les membres de votre personnel qui ont besoin d'accéder à vos applications cloud Comptes AWS ou à celles de celles-ci. Pour permettre aux utilisateurs d'accéder à vos CodeCommit référentiels sans configurer les utilisateurs IAM, vous pouvez configurer l'git-remote-codecommitutilitaire. Pour plus d'informations

sur IAM et CodeCommit, consultezUtilisation d'IAM avec CodeCommit : informations d'identification Git, clés SSH et AWS clés d'accès. Pour plus d'informations sur la configuration de l'git-remote-codecommitutilitaire, consultez la section Connexion aux AWS CodeCommit référentiels avec des informations d'identification rotatives dans le Guide de AWS CodeCommit l'utilisateur.

- Amazon Keyspaces (for Apache Cassandra) access (Accès Amazon Keyspaces (pour Apache Cassandra)) : dans une situation où vous n'êtes pas en mesure de vous servir des utilisateurs dans IAM Identity Center, par exemple à des fins de test de compatibilité avec Cassandra, vous pouvez utiliser un utilisateur IAM avec des informations d'identification spécifiques au service pour vous authentifier auprès d'Amazon Keyspaces. Les utilisateurs d'IAM Identity Center sont les membres de votre personnel qui ont besoin d'accéder à vos applications cloud Comptes AWS ou à celles de celles-ci. Vous pouvez également vous connecter à Amazon Keyspaces à l'aide d'informations d'identification temporaires. Pour de plus d'informations, veuillez consulter Utilisation d'informations d'identification temporaires pour se connecter à Amazon Keyspaces à l'aide d'un rôle IAM et du plugin Sigv4 dans le Guide du développeur Amazon Keyspaces (pour Apache Cassandra).

Suivez les meilleures pratiques pour protéger vos informations d'identification de l'utilisateur root

Lorsque vous créez un Compte AWS, vous définissez les informations d'identification de l'utilisateur root pour vous connecter au AWS Management Console. Protégez vos informations d'identification d'utilisateur root de la même manière que vous protégez d'autres informations personnelles sensibles. Pour mieux comprendre comment sécuriser et mettre à l'échelle vos processus d'utilisateur root, consultez Bonnes pratiques d'utilisateur root pour votre Compte AWS.

Accorder les autorisations de moindre privilège

Lorsque vous définissez des autorisations avec des stratégies IAM, accordez uniquement les autorisations nécessaires à l'exécution d'une seule tâche. Pour ce faire, vous définissez les actions qui peuvent être entreprises sur des ressources spécifiques dans des conditions spécifiques, également appelées autorisations de moindre privilège. Vous pouvez commencer par des autorisations étendues tout en explorant les autorisations requises pour votre charge de travail ou votre cas d'utilisation. Au fur et à mesure que votre cas d'utilisation évolue, vous pouvez réduire les autorisations que vous accordez pour obtenir le moindre privilège. Pour de plus d'informations sur l'utilisation de IAM pour appliquer des autorisations, consultez Politiques et autorisations dans IAM.

Commencez avec les politiques AWS gérées et passez aux autorisations du moindre privilège

Pour commencer à accorder des autorisations à vos utilisateurs et charges de travail, utilisez les AWS politiques gérées qui accordent des autorisations dans de nombreux cas d'utilisation courants. Ils sont disponibles dans votre Compte AWS. N'oubliez pas que les politiques AWS gérées peuvent ne pas accorder d'autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles peuvent être utilisées par tous les AWS clients. En conséquence, nous vous recommandons de réduire encore les autorisations en définissant des Politiques gérées par le client qui sont spécifiques à vos cas d'utilisation. Pour plus d'informations, consultez AWS politiques gérées. Pour plus d'informations sur les politiques AWS gérées conçues pour des fonctions professionnelles spécifiques, consultezAWS politiques gérées pour les fonctions professionnelles.

Utiliser IAM Access Analyzer pour générer des politiques de moindre privilège en fonction de l'activité d'accès

Pour accorder uniquement les autorisations nécessaires à l'exécution d'une seule tâche, vous pouvez générer des politiques en fonction de votre activité d'accès connectée AWS CloudTrail. IAM Access Analyzer analyse les services et les actions que vos rôles IAM utilisent, puis génère une stratégie précise que vous pouvez utiliser. Après avoir testé chaque stratégie générée, vous pouvez la déployer dans votre environnement de production. Cela garantit que vous n'accordez que les autorisations requises à vos charges de travail. Pour plus d'informations sur la génération de politiques, consultez IAM Access Analyzer policy generation.

Vérifiez et supprimez régulièrement les utilisateurs, les rôles, les autorisations, les politiques et les informations d'identification non utilisés

Vous avez peut-être des utilisateurs, des rôles, des autorisations, des stratégies ou des informations d'identification IAM dont vous n'avez plus besoin dans votre Compte AWS. IAM fournit dernière information consultées pour vous aider à identifier les utilisateurs, les rôles, les autorisations, les stratégies et les informations d'identification dont vous n'avez plus besoin afin de pouvoir les supprimer. Cela vous permet de réduire le nombre d'utilisateurs, de rôles, d'autorisations, de politiques et d'informations d'identification que vous devez surveiller. Vous pouvez ainsi peaufiner vos politiques IAM afin qu'elles respectent au plus près le principe du moindre privilège. Pour plus d'informations, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Utiliser des conditions dans les stratégies IAM pour restreindre davantage l'accès

Vous pouvez spécifier les conditions dans lesquelles une déclaration de politique est en vigueur. Ainsi, vous pouvez accorder l'accès aux actions et aux ressources, mais uniquement si la demande d'accès répond à des conditions spécifiques. Par exemple, vous pouvez écrire une condition de stratégie pour spécifier que toutes les demandes doivent être envoyées via SSL. Vous pouvez également utiliser des conditions pour accorder l'accès aux actions de service, mais uniquement si elles sont utilisées par le biais d'un service spécifique Service AWS, tel que AWS CloudFormation. Pour plus d'informations, consultez Éléments de politique JSON IAM : Condition.

Vérifiez l'accès public et intercompte aux ressources avec IAM Access Analyzer.

Avant d'accorder des autorisations d'accès public ou multicompte AWS, nous vous recommandons de vérifier si un tel accès est requis. Vous pouvez utiliser IAM Access Analyzer pour vous aider à prévisualiser et à analyser l'accès public et intercompte pour les types de ressources pris en charge. Pour ce faire, consultez les résultats que génère IAM Access Analyzer. Ces résultats vous aident à vérifier que vos contrôles d'accès aux ressources accordent l'accès que vous attendez. En outre, lorsque vous mettez à jour les autorisations publiques et entre comptes, vous pouvez vérifier l'effet de vos modifications avant de déployer de nouveaux contrôles d'accès à vos ressources. IAM Access Analyzer surveille également en permanence les types de ressources pris en charge et génère une recherche pour les ressources qui autorisent un accès public ou intercompte. Pour plus d'informations, consultez l'onglet Prévisualisation de l'accès avec les API IAM Access Analyzer.

Utilisez IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles

Validez les politiques que vous créez pour vous assurer qu'elles respectent le langage de politique IAM (JSON) et les bonnes pratiques IAM. Vous pouvez valider vos politiques à l'aide de la vérification de politique IAM Access Analyzer. IAM Access Analyzer fournit plus de 100 vérifications de politiques et des recommandations exploitables pour vous aider à créer des politiques sécurisées et fonctionnelles. Lorsque vous créez de nouvelles politiques ou que vous modifiez des politiques existantes dans la console, IAM Access Analyzer fournit des recommandations pour vous aider à affiner et à valider vos politiques avant de les enregistrer. En outre, nous vous recommandons d'examiner et de valider toutes vos politiques existantes. Pour de plus d'informations,

veuillez consulter l'onglet Validation de politique IAM Access Analyzer. Pour en savoir plus sur les vérifications de politique fournies par IAM Access Analyzer, veuillez consulter IAM Access Analyzer policy validation (Validation de politique IAM Access Analyzer).

Établir des garde-fous d'autorisations sur plusieurs comptes

Lorsque vous augmentez vos charges de travail, séparez-les en utilisant plusieurs comptes gérés avec AWS Organizations. Nous vous recommandons d'utiliser Organizations Politiques de contrôle des services (SCP) pour établir des garde-fous d'autorisations afin de contrôler l'accès de tous les utilisateurs et rôles IAM de vos comptes. Les SCP sont un type de politique d'organisation que vous pouvez utiliser pour gérer les autorisations dans votre organisation au niveau de l' AWS organisation, de l'unité d'organisation ou du compte. Les garde-fous d'autorisations que vous établissez s'appliquent à tous les utilisateurs et rôles au sein des comptes couverts. Les politiques de contrôle des services ne suffisent cependant pas à accorder des autorisations aux comptes de votre organisation. Pour ce faire, votre administrateur doit toujours attacher des politiques basées sur les ressources ou basées sur l'identité aux utilisateurs ou aux rôles IAM, ou aux ressources de vos comptes pour accorder réellement des autorisations. Pour de plus d'informations, veuillez consulter AWS Organizations, comptes et garde-corps IAM.

Utiliser les limites des autorisations pour déléguer la gestion des autorisations au sein d'un compte.

Dans certains scénarios, vous souhaitez peut-être déléguer la gestion des autorisations d'un compte à d'autres. Par exemple, vous pouvez autoriser les développeurs à créer et à gérer des rôles pour leurs charges de travail. Lorsque vous déléguez des autorisations à d'autres personnes, utilisez Limites d'autorisations pour définir les autorisations maximales que vous déléguez. Une limite d'autorisations est une fonctionnalité avancée permettant d'utiliser une politique gérée pour définir les autorisations maximales qu'une politique basée sur l'identité peut accorder à une rôle IAM. Une limite d'autorisations restreint le nombre maximum d'autorisations, mais n'accorde pas seule l'accès. Pour plus d'informations, voir Limites d'autorisations pour les entités IAM.

Bonnes pratiques d'utilisateur root pour votre Compte AWS

Lorsque vous créez un Compte AWS, vous commencez par un ensemble d'informations d'identification par défaut avec un accès complet à toutes les AWS ressources de votre compte. Cette identité est appelée l'utilisateur root Compte AWS. Nous vous recommandons vivement de ne pas accéder à l'utilisateur Compte AWS root, sauf si vous avez une tâche qui nécessite des

informations d'identification de l'utilisateur root. Vous devez sécuriser vos informations d'identification de l'utilisateur root et les mécanismes de récupération de votre compte afin de ne pas exposer vos informations d'identification hautement privilégiées à des fins d'utilisation non autorisée.

Au lieu d'accéder à l'utilisateur root, créez un utilisateur administratif pour les tâches quotidiennes.

- Pour un single autonome Compte AWS, voirCréation d'un utilisateur doté d'un accès administratif.
- Pour plusieurs sites Comptes AWS gérés via AWS Organizations, voir Configurer Compte AWS l'accès pour un utilisateur administratif d'IAM Identity Center.

Avec votre utilisateur administratif, vous pouvez ensuite créer des identités supplémentaires pour les utilisateurs qui ont besoin d'accéder aux ressources de votre Compte AWS. Nous vous recommandons vivement de demander aux utilisateurs de s'authentifier avec des informations d'identification temporaires lors de l'accès AWS.

- Pour un compte unique et autonome Compte AWS, utilisez-le Rôles IAM pour créer des identités dans votre compte avec des autorisations spécifiques. Les rôles sont destinés à être endossés par toute personne qui en a besoin. En outre, un rôle ne dispose pas d'informations d'identification standard à long terme comme un mot de passe ou des clés d'accès associées. Au lieu de cela, lorsque vous adoptez un rôle, il vous fournit des informations d'identification de sécurité temporaires pour votre session de rôle. Contrairement aux rôles IAM, Utilisateurs IAM dispose d'informations d'identification à long terme telles que des mots de passe et des clés d'accès. Dans la mesure du possible, nous vous recommandons, dans le cadre des bonnes pratiques, de vous appuyer sur des informations d'identification temporaires plutôt que de créer des utilisateurs IAM ayant des informations d'identification à long terme tels que les clés d'accès.
- Pour plusieurs organisations Comptes AWS gérées par le biais d'Organizations, utilisez les utilisateurs du personnel d'IAM Identity Center. Avec IAM Identity Center, vous pouvez gérer de manière centralisée les utilisateurs de vos comptes Comptes AWS et les autorisations associées à ces comptes. Gérez vos identités d'utilisateur à l'aide d'IAM Identity Center ou depuis un fournisseur d'identité externe. Pour plus d’informations, consultez Présentation de AWS IAM Identity Center dans le Guide de l’utilisateur AWS IAM Identity Center .

Rubriques

- Sécurisez vos informations d’identification d'utilisateur root pour empêcher toute utilisation non autorisée
- Utiliser un mot de passe utilisateur root fort pour protéger l'accès

- Sécurisez votre connexion d'utilisateur root avec l'authentification multifactorielle (MFA)
- Ne créer aucune clé d'accès pour l'utilisateur root
- Utiliser une approbation par plusieurs personnes pour la connexion de l'utilisateur root dans la mesure du possible
- Utiliser une adresse e-mail de groupe pour les informations d'identification de l'utilisateur root
- Restreindre l'accès aux mécanismes de récupération des comptes
- Sécurisez les informations d’identification d'utilisateur root de votre compte Organizations
- Surveiller l'accès et l'utilisation

Sécurisez vos informations d’identification d'utilisateur root pour empêcher toute utilisation non autorisée

Sécurisez vos informations d’identification d'utilisateur root et ne les utilisez que pour les tâches qui les nécessitent. Pour empêcher toute utilisation non autorisée, ne partagez pas le mot de passe de votre utilisateur root, votre MFA, vos clés d'accès, vos paires de clés ou vos certificats de signature avec qui que ce soit, à l'exception de ceux qui ont un besoin professionnel strict d'accéder à l'utilisateur root. CloudFront

Ne stockez pas le mot de passe de l'utilisateur root Services AWS dans des outils qui dépendent d'un compte auquel on accède avec ce même mot de passe. Si vous perdez ou oubliez votre mot de passe utilisateur root, vous ne pourrez pas accéder à ces outils. Nous vous recommandons de donner la priorité à la résilience et d'envisager de demander à deux personnes ou plus d'autoriser l'accès à l'emplacement de stockage. L'accès au mot de passe ou à son emplacement de stockage doit être consigné et surveillé.

Utiliser un mot de passe utilisateur root fort pour protéger l'accès

Nous vous recommandons d'utiliser un mot de passe fort et unique. Des outils tels que des gestionnaires de mots de passe dotés d'algorithmes de génération de mots de passe puissants peuvent vous permettre d'atteindre ces objectifs. AWS exige que votre mot de passe remplisse les conditions suivantes :

- Avoir un minimum de 8 caractères et un maximum de 128 caractères
- Inclure au minimum trois des types de caractères suivants : majuscules, minuscules, chiffres, et les symboles ! @ # $ % ^ & * () <> [] {} | _ + - =

- Il ne doit pas être identique à votre Compte AWS nom ou à votre adresse e-mail.

Pour plus d'informations, consultez Modifiez le mot de passe du Utilisateur racine d'un compte AWS.

Sécurisez votre connexion d'utilisateur root avec l'authentification multifactorielle (MFA)

Étant donné qu'un utilisateur root peut effectuer des actions privilégiées, il est essentiel d'ajouter MFA pour l'utilisateur root comme deuxième facteur d'authentification, en plus de l'adresse e-mail et du mot de passe comme informations d'identification de connexion. Nous vous recommandons fortement d'activer plusieurs MFA pour vos informations d'identification d'utilisateur root afin de renforcer la flexibilité et la résilience de votre stratégie de sécurité. Vous pouvez enregistrer jusqu'à huit dispositifs MFA de n'importe quelle combinaison de types MFA actuellement pris en charge avec l'utilisateur root de votre Compte AWS .

- Les clés de sécurité matérielles certifiées FIDO sont fournies par des fournisseurs tiers. Pour plus d'informations, voir Activer une clé de sécurité FIDO pour l'utilisateur Compte AWS root.
- Un périphérique matériel qui génère un code numérique à six chiffres basé sur l'algorithme TOTP (mot de passe unique à durée limitée). Pour plus d'informations, voir Activer un jeton TOTP matériel pour l'utilisateur Compte AWS root.
- Une appli d'authentification virtuelle qui s'exécute sur un téléphone ou un autre appareil et qui égale le niveau d'un dispositif physique. Pour plus d'informations, voir Activer un périphérique MFA virtuel pour votre Compte AWS utilisateur root.

Ne créer aucune clé d'accès pour l'utilisateur root

Les touches d'accès vous permettent d'exécuter des commandes dans l'interface de ligne de AWS commande (AWS CLI) ou d'utiliser des opérations d'API à partir de l'un des AWS SDK. Nous vous recommandons vivement de ne pas créer de paires de clés d'accès pour votre utilisateur root, car celui-ci dispose d'un accès complet à toutes Services AWS les ressources du compte, y compris aux informations de facturation.

Étant donné que seules quelques tâches nécessitent l'utilisateur root et que vous les effectuez généralement rarement, nous vous recommandons de vous connecter au pour effectuer des AWS Management Console tâches d'utilisateur root. Avant de créer des clés d'accès, passez en revue les alternatives aux clés d'accès à long terme.

Utiliser une approbation par plusieurs personnes pour la connexion de l'utilisateur root dans la mesure du possible

Envisagez d'utiliser l'approbation par plusieurs personnes pour garantir qu'aucune personne ne puisse accéder à la fois à la MFA et au mot de passe de l'utilisateur root. Certaines entreprises ajoutent une couche de sécurité supplémentaire en configurant un groupe d'administrateurs ayant accès au mot de passe et un autre groupe d'administrateurs ayant accès à la MFA. Un membre de chaque groupe doit se réunir pour se connecter en tant qu'utilisateur root.

Utiliser une adresse e-mail de groupe pour les informations d'identification de l'utilisateur root

Utilisez une adresse e-mail gérée par votre entreprise et transférez les messages reçus directement à un groupe d'utilisateurs. Si vous AWS devez contacter le titulaire du compte, cette approche réduit le risque de retard dans la réponse, même si les personnes sont en vacances, sont malades ou ont quitté l'entreprise. L'adresse e-mail utilisée pour l'utilisateur root ne doit pas être utilisée à d'autres fins.

Restreindre l'accès aux mécanismes de récupération des comptes

Veillez à développer un processus pour gérer les mécanismes de récupération des informations d'identification d'utilisateur root au cas où vous auriez besoin d'y accéder en cas d'urgence, telle que la prise de contrôle de votre compte administratif.

- Assurez-vous d'avoir accès à votre boîte de réception de messagerie d'utilisateur root afin de pouvoir réinitialiser le mot de passe d'utilisateur root perdu ou oublié.
- Si le MFA de votre utilisateur Compte AWS root est perdu, endommagé ou ne fonctionne pas, vous pouvez vous connecter à l'aide d'un autre MFA enregistré avec les mêmes informations d'identification d'utilisateur root. Si vous avez perdu l'accès à tous vos MFA, vous avez besoin du numéro de téléphone et de l'e-mail utilisés pour enregistrer votre compte, pour être à jour et accessible afin de récupérer votre MFA. Pour de plus amples informations, consultez la section Récupération d'un dispositif MFA d'utilisateur root.
- Si vous choisissez de ne pas stocker votre mot de passe d'utilisateur root et votre MFA, le numéro de téléphone enregistré dans votre compte peut être utilisé comme autre moyen de récupérer les informations d'identification d'utilisateur root. Assurez-vous d'avoir accès au numéro de téléphone de contact, maintenez-le à jour et limitez le nombre de personnes autorisées à gérer le numéro de téléphone.

Personne ne doit avoir accès à la fois à la boîte de réception de messagerie et au numéro de téléphone, car les deux sont des canaux de vérification permettant de récupérer votre mot de passe d'utilisateur root. Il est important que deux groupes de personnes gèrent ces canaux. Un groupe ayant accès à votre adresse e-mail principale et un autre groupe ayant accès au numéro de téléphone principal pour récupérer l'accès à votre compte en tant qu'utilisateur root.

Sécurisez les informations d’identification d'utilisateur root de votre compte Organizations

Au fur et à mesure que vous passez à une stratégie multi-comptes avec Organizations, chacun de vos utilisateurs Comptes AWS possède ses propres informations d'identification d'utilisateur root que vous devez sécuriser. Le compte que vous utilisez pour créer votre organisation est le compte de gestion et les autres comptes de votre organisation sont des comptes membres.

Sécuriser les informations d'identification d'utilisateur root pour les comptes membres

Si vous utilisez Organizations pour gérer plusieurs comptes, vous pouvez adopter deux stratégies pour sécuriser l'accès de l'utilisateur root dans vos Organisations.

- Sécurisez les informations d’identification d'utilisateur root de vos comptes des Organizations avec MFA.
- Ne réinitialisez pas le mot de passe de l'utilisateur root pour vos comptes et récupérez l'accès à celui-ci uniquement lorsque cela est nécessaire en utilisant le processus de réinitialisation du mot de passe. Lorsque vous créez un compte membre dans votre organisation, Organizations crée automatiquement un rôle IAM dans le compte membre qui permet au compte de gestion d'accéder temporairement au compte membre.

Pour plus de détails, consultez la section Accès aux comptes membres de votre organisation dans le Guide de l'utilisateur Organizations.

Définir des contrôles de sécurité préventifs dans Organisations à l'aide d'une politique de contrôle des services (SCP)

Si vous utilisez Organizations pour gérer plusieurs comptes, vous pouvez appliquer une SCP pour restreindre l'accès à l'utilisateur root des comptes membres. Le fait de refuser toutes les actions de l'utilisateur root sur vos comptes membres, à l'exception de certaines actions réservées au root, permet d'empêcher tout accès non autorisé. Pour plus de détails, consultez la section Utiliser une

politique de contrôle des services (SCP) pour restreindre ce que l'utilisateur root de vos comptes membres peut faire.

Surveiller l'accès et l'utilisation

Nous vous recommandons d'utiliser vos mécanismes de suivi actuels pour surveiller, alerter et signaler la connexion et l'utilisation des informations d'identification d'utilisateur root, y compris les alertes annonçant la connexion et l'utilisation de l'utilisateur root. Les services suivants peuvent aider à garantir le suivi de l'utilisation des informations d'identification d'utilisateur root et à effectuer des contrôles de sécurité afin d'empêcher toute utilisation non autorisée.

- Si vous souhaitez être informé de l'activité de connexion de l'utilisateur root sur votre compte, vous pouvez utiliser Amazon CloudWatch pour créer une règle d'événements qui détecte l'utilisation des informations d'identification de l'utilisateur root et déclenche une notification destinée à votre administrateur de sécurité. Pour plus de détails, voir Surveillance et notification de l'activité de l'utilisateur Compte AWS root.
- Si vous souhaitez configurer des notifications pour vous avertir des actions approuvées de l'utilisateur root, vous pouvez utiliser Amazon et Amazon EventBridge SNS pour rédiger une EventBridge règle permettant de suivre l'utilisation de l'utilisateur root pour l'action spécifique et de vous en informer via une rubrique Amazon SNS. Pour obtenir un exemple, consultez Envoyer une notification lorsqu'un objet Amazon S3 est créé.
- Si vous l'utilisez déjà GuardDuty comme service de détection des menaces, vous pouvez étendre sa capacité à vous avertir lorsque les informations d'identification de l'utilisateur root sont utilisées dans votre compte.

Cette alerte doit inclure, sans s'y limiter, l'adresse e-mail pour l'utilisateur root. Vérifiez que vous avez des procédures en place pour savoir comment répondre aux alertes afin que le personnel qui reçoit une alerte d'accès d'utilisateur root comprenne comment confirmer que l'accès de l'utilisateur root est attendu et comment remonter l'événement s'il croit qu'un incident de sécurité est en cours. Pour obtenir un exemple illustrant la configuration des alertes, consultez Surveiller et notifier l'activité de l'utilisateur root du Compte AWS.

Évaluer la conformité de la MFA de l'utilisateur root

- AWS Config utilise des règles pour aider à appliquer les meilleures pratiques pour les utilisateurs root. Vous pouvez utiliser des règles AWS gérées pour obliger les utilisateurs root à activer

l'authentification multifactorielle (MFA). AWS Config peut également identifier les clés d'accès pour l'utilisateur root.

- Security Hub vous fournit une vue complète de l'état de votre sécurité AWS et vous aide à évaluer votre AWS environnement par rapport aux normes et aux meilleures pratiques du secteur de la sécurité, telles que l'utilisation du MFA sur l'utilisateur root et l'absence de clés d'accès utilisateur root. Pour plus de détails sur les règles disponibles, consultez Contrôles AWS Identity and Access Management dans le Guide de l'utilisateur de Security Hub.
- Trusted Advisor fournit un contrôle de sécurité qui vous permet de savoir si le MFA n'est pas activé sur le compte utilisateur root. Pour de plus amples informations, consultez MFA pour un compte root dans le Guide de l'utilisateur AWS .

Si vous devez signaler un problème de sécurité lié à votre compte, consultez Signaler des e-mails suspects ou Signaler une vulnérabilité. Vous pouvez également contacter AWS pour obtenir de l'aide et des conseils supplémentaires.

Cas d'utilisation métier pour IAM

Un exemple d'utilisation commerciale simple de l'IAM peut vous aider à comprendre les méthodes de base que vous pouvez utiliser pour implémenter le service afin de contrôler l' AWS accès de vos utilisateurs. Celui-ci est décrit dans des termes généraux, sans les mécanismes d'utilisation de l'API IAM pour obtenir les résultats escomptés.

Ce cas d'utilisation présente deux manières standard dont une entreprise fictive appelée Example Corp peut utiliser IAM. Le premier scénario prend en compte Amazon Elastic Compute Cloud (Amazon EC2). Le second prend en compte Amazon Simple Storage Service (Amazon S3).

Pour plus d'informations sur l'utilisation d'IAM avec d'autres services de AWS, consultezAWS services qui fonctionnent avec IAM.

Rubriques

- Configuration initiale d'Example Corp
- Cas d'utilisation pour IAM avec Amazon EC2
- Cas d'utilisation pour IAM avec Amazon S3

Configuration initiale d'Example Corp

Nikki Wolf et Mateo Jackson sont les fondateurs d'Example Corp. Au démarrage de l'entreprise, ils créent un Compte AWS and AWS IAM Identity Center(IAM Identity Center) pour créer des comptes administratifs à utiliser avec leurs ressources. AWS Lorsque vous configurez l'accès au compte pour l'utilisateur administratif, IAM Identity Center crée un rôle IAM correspondant. Ce rôle, qui est contrôlé par IAM Identity Center, est créé dans le répertoire approprié Compte AWS, et les politiques spécifiées dans le jeu d'AdministratorAccessautorisations sont associées au rôle.

Comme ils ont désormais des comptes administrateurs, Nikki et Mateo n'ont plus besoin d'utiliser leur utilisateur root pour accéder à leur Compte AWS. Ils ne prévoient de l'utiliser que pour effectuer les tâches que lui seul peut effectuer. Après avoir passé en revue les bonnes pratiques de sécurité, ils configurent une authentification multifactorielle (MFA) pour leurs informations d'identification d'utilisateur root et décident de la manière de protéger ces informations d'identification d'utilisateur root.

Au fur et à mesure où leur entreprise croît, ils embauchent des employés comme développeurs, administrateurs, testeurs, gestionnaires et administrateurs système. Nikki est en charge des opérations, tandis que Mateo gère les équipes d'ingénierie. Ils ont mis en place un serveur de domaine Active Directory pour gérer les comptes des employés et gérer l'accès aux ressources internes de l'entreprise.

Pour permettre à leurs employés d'accéder aux AWS ressources, ils utilisent IAM Identity Center pour connecter l'Active Directory de leur entreprise à leur Compte AWS.

Comme ils ont connecté Active Directory à IAM Identity Center, les utilisateurs, les groupes et les appartenances aux groupes sont synchronisés et définis. Ils doivent attribuer des ensembles d'autorisations et des rôles aux différents groupes afin de donner aux utilisateurs le niveau d'accès approprié aux AWS ressources. Ils utilisent AWS politiques gérées pour les fonctions professionnelles in AWS Management Console pour créer ces ensembles d'autorisations :

- Administrateur
- Facturation
- Développeurs
- Administrateurs réseau
- Administrateurs de base de données
- Administrateurs système

- Utilisateurs du support

Ils attribuent ensuite ces ensembles d'autorisations aux rôles attribués à leurs groupes Active Directory.

Pour un step-by-step guide décrivant la configuration initiale d'IAM Identity Center, voir Getting started dans le guide de l'AWS IAM Identity Center utilisateur. Pour plus d'informations sur l'approvisionnement de l'accès utilisateur à IAM Identity Center, consultez Accès par authentification unique aux comptes AWS dans le Guide de l'utilisateur AWS IAM Identity Center .

Cas d'utilisation pour IAM avec Amazon EC2

Une entreprise comme Example Corp utilise généralement IAM pour interagir avec des services comme Amazon EC2. Pour comprendre cette partie du cas d'utilisation, vous devez comprendre les bases d'Amazon EC2. Pour plus d'informations sur Amazon EC2, consultez le guide de l'utilisateur Amazon EC2.

Autorisations Amazon EC2 pour les groupes d'utilisateurs

Pour fournir un contrôle du « périmètre », Nikki attache une politique au groupe AllUsers d'utilisateurs. Cette politique refuse toute AWS demande d'un utilisateur si l'adresse IP d'origine se trouve en dehors du réseau d'entreprise d'Example Corp.

Chez Example Corp, différents groupes d'utilisateurs nécessitent des autorisations distinctes :

- Administrateurs système : nécessitent l'autorisation de créer et de gérer des AMI, des instances, des instantanés, des volumes, des groupes de sécurité, etc. Nikki attache la politique `AmazonEC2FullAccess` AWS gérée au groupe SysAdmins d'utilisateurs qui autorise les membres du groupe à utiliser toutes les actions Amazon EC2.
- Développeurs : ont uniquement besoin de pouvoir utiliser des instances. Par conséquent, Mateo crée et attache une politique au groupe d'utilisateurs Développeurs qui autorise les développeurs à appeler `DescribeInstances`, `RunInstances`, `StopInstances`, `StartInstances` et `TerminateInstances`.

> **Note**
>
> Amazon EC2 utilise des clés SSH, des mots de passe Windows et des groupes de sécurité pour contrôler qui a accès au système d'exploitation d'instances Amazon EC2 spécifiques.

Il n'existe aucune méthode dans le système IAM pour autoriser ou refuser l'accès au système d'exploitation d'une instance spécifique.

- Utilisateurs du support : ne doivent pas pouvoir exécuter toutes les actions Amazon EC2, uniquement répertorier les ressources Amazon EC2 disponibles actuellement. Par conséquent, Nikki crée et attache une politique au groupe d'utilisateurs Utilisateurs du support qui les autorise uniquement à appeler des opérations d'API « Describe » Amazon EC2.

Pour des exemples de ce à quoi peuvent ressembler ces politiques, consultez Exemples de politiques basées sur l'identité IAM la section Using AWS Identity and Access Management du guide de l'utilisateur Amazon EC2.

Modification de la fonction de tâche de l'utilisateur

À un moment donné, l'un des développeurs, Paulo Santos, change de rôle et devient gestionnaire. En tant que gestionnaire, Paulo fait partie du groupe Utilisateurs du support afin de pouvoir ouvrir des dossiers d'assistance pour ses développeurs. Mateo déplace Paulo du groupe Développeurs vers le groupe d'utilisateurs Utilisateurs du support. En conséquence de ce déplacement, sa capacité à interagir avec des instances Amazon EC2 est limitée. Il ne peut pas lancer ou démarrer des instances. Il ne peut également pas arrêter ou mettre hors service les instances existantes, même s'il était l'utilisateur qui a lancé ou démarré l'instance. Il peut uniquement répertorier les instances lancées par les utilisateurs d'Example Corp.

Cas d'utilisation pour IAM avec Amazon S3

Les entreprises comme Example Corp utilisent aussi généralement IAM avec Amazon S3. John a créé un compartiment Amazon S3 appelé aws-s3-bucket pour l'entreprise.

Création d'autres utilisateurs et groupes d'utilisateurs

En tant qu'employés, Zhang Wei et Mary Major ont besoin de créer chacun leurs propres données dans le compartiment de l'entreprise. Ils ont également besoin des données partagées en lecture et en écriture que tous les développeurs utiliseront. Pour cela, Mateo organise logiquement les données dans aws-s3-bucket à l'aide d'un schéma de préfixe de clé Amazon S3 présenté dans l'illustration suivante.

```
/aws-s3-bucket
    /home
```

```
        /zhang
        /major
    /share
        /developers
        /managers
```

Mateo divise `/aws-s3-bucket` en un ensemble de répertoires de base pour chaque employé et une zone partagée de groupes de développeurs et de gestionnaires.

À présent, Mateo crée un ensemble de politiques pour attribuer des autorisations aux utilisateurs et aux groupes d'utilisateurs :

- Accès au répertoire de base de Zhang : Mateo attache une politique à Wei lui permettant de lire, d'écrire et de répertorier les objets avec le préfixe de clé Amazon S3 `/aws-s3-bucket/home/zhang/`
- Accès au répertoire de base de Major : Mateo attache une politique à Mary lui permettant de lire, d'écrire et de répertorier les objets avec le préfixe de clé Amazon S3 `/aws-s3-bucket/home/major/`
- Accès au répertoire partagé par le groupe d'utilisateurs Développeurs : Mateo attache une politique au groupe qui permet aux développeurs de lire, d'écrire et de répertorier tous les objets de `/aws-s3-bucket/share/developers/`
- Accès au répertoire partagé par le groupe d'utilisateurs Gestionnaires : Mateo attache une politique au groupe d'utilisateurs qui permet aux gestionnaires de lire, d'écrire et de répertorier tous les objets de `/aws-s3-bucket/share/managers/`

> **Note**
>
> Amazon S3 n'attribue pas automatiquement à un utilisateur qui crée un compartiment ou un objet l'autorisation d'exécuter d'autres actions sur ce compartiment ou cet objet. Par conséquent, dans vos politiques IAM, vous devez accorder explicitement l'autorisation aux utilisateurs d'utiliser les ressources Amazon S3 qu'ils créent.

Pour obtenir des exemples de ce à quoi ces politiques peuvent ressembler, veuillez consulter Access Control (contrôle d'accès) dans le guide de l'utilisateur service de stockage simple Amazon. Pour plus d'informations sur la manière dont les politiques sont évaluées au moment de l'exécution, consultez Logique d'évaluation de politiques.

Modification de la fonction de tâche de l'utilisateur

À un moment donné, l'un des développeurs, Zhang Wei, change de rôle et devient gestionnaire. Nous supposerons qu'il n'a plus besoin d'accéder aux documents de l'annuaire `share/developers`. Mateo, en tant qu'administrateur, déplace Wei du groupe d'utilisateurs `Managers` vers le groupe d'utilisateurs `Developers`. En une simple réaffectation, Wei obtient automatiquement toutes les autorisations accordées au groupe d'utilisateurs `Managers`, mais il en peut plus accéder aux données du répertoire `share/developers`.

Intégration d'une entreprise tierce

Les organisations travaillent souvent avec des entreprises partenaires, des consultants et des sous-traitants. Example Corp a un partenaire appelé Widget Company, et une employée de Widget Company appelé Shirley Rodriguez doit placer des données dans un compartiment à l'intention d'Example Corp. Nikki crée un groupe d'utilisateurs appelé WidgetCoet un nom d'utilisateur `Shirley` et ajoute Shirley au groupe WidgetCo d'utilisateurs. Nikki crée également un compartiment spécial appelé aws-s3-bucket1 pour Shirley.

Nikki met à jour les politiques existantes ou ajoute de nouvelles politiques pour le partenaire Widget Company. Par exemple, Nikki peut créer une nouvelle politique qui empêche les membres du groupe WidgetCo d'utilisateurs d'utiliser des actions autres que l'écriture. Cette politique est nécessaire uniquement s'il existe une vaste politique qui donne à tous les utilisateurs l'accès à un éventail d'actions Amazon S3.

Didacticiels IAM

Les didacticiels suivants présentent end-to-end des procédures complètes pour les tâches courantes pour AWS Identity and Access Management (IAM). Ils sont conçus pour une environnement de travaux pratiques, avec des exemples fictifs de noms de sociétés, de noms d'utilisateur, etc. Leur objectif consiste à fournir des instructions générales. Ils ne sont pas conçus pour être utilisés directement dans votre environnement de production sans être préalablement vérifiés et adaptés soigneusement aux besoins uniques de votre organisation.

Didacticiels

- Tutoriel IAM : Accorder l'accès à la console de facturation
- Didacticiel IAM : déléguer l'accès entre des comptes AWS à l'aide des rôles IAM
- Didacticiel IAM : créer et attacher votre première politique gérée par le client
- Tutoriel IAM : définir les autorisations d'accès aux AWS ressources en fonction des balises
- Didacticiel IAM : permettre aux utilisateurs de gérer leurs informations d'identification et leurs paramètres MFA

Tutoriel IAM : Accorder l'accès à la console de facturation

Le Compte AWS propriétaire (Utilisateur racine d'un compte AWS) peut accorder aux utilisateurs et aux rôles IAM l'accès aux AWS Billing and Cost Management données les Compte AWS concernant. Les instructions de ce tutoriel vous aident à configurer un scénario prétesté. Ce scénario vous aide à acquérir une expérience pratique de la configuration des autorisations de facturation sans avoir à vous préoccuper de votre compte AWS de production principal.

Prérequis

Effectuez les opérations suivantes avant de commencer à suivre les étapes de ce tutoriel :

- Créez un test Compte AWS.
- Connectez-vous à votre test en Compte AWS tant qu'utilisateur root.
- Enregistrez le Compte AWS numéro de votre compte de test afin de pouvoir l'utiliser dans le didacticiel. Par exemple, dans ce tutoriel, nous utilisons le numéro de compte 111122223333. Chaque fois qu'une étape utilise ce numéro de compte, remplacez-le par votre numéro de compte de test.

Étape 1 : Activez l'accès IAM aux informations de facturation sur votre compte de test AWS

Dans ce scénario, vous vous connectez à votre test en Compte AWS tant qu'utilisateur root pour accorder à IAM l'accès aux informations de facturation. Lorsque vous accordez à IAM l'accès aux informations de facturation, les utilisateurs et les rôles IAM peuvent accéder à la AWS Billing and Cost Management console. Ce paramètre n'accorde pas aux utilisateurs et aux rôles IAM les autorisations nécessaires pour accéder à ces pages de la console. Il accorde l'accès aux utilisateurs ou aux rôles IAM qui disposent des politiques IAM requises. Si des politiques sont déjà associées à des utilisateurs ou à des rôles IAM, mais que ce paramètre n'est pas activé, les autorisations accordées par ces politiques ne sont pas en vigueur.

> **Note**
>
> Comptes AWS créé à l'aide AWS Organizations de l'activation par défaut de l'accès IAM aux informations de facturation.

Étape 2 : Créer des utilisateurs et des groupes de test

Dans ce scénario, vous accordez aux utilisateurs IAM l'accès à la console de facturation et vous créez deux utilisateurs :

- Pat Candella

  Pat est membre du département financier et s'occupe de la facturation et des paiements. Pat a besoin d'un accès complet aux informations de facturation figurant dans votre Compte AWS.

- Terry Whitlock

  Terry fait partie de votre service d'assistance informatique. La plupart du temps, Terry n'a pas besoin d'accéder à la console de facturation, mais il en a parfois besoin pour répondre aux questions des employés du département financier.

Étape 3 : Créer un rôle pour accorder l'accès à la console AWS Billing

Un rôle IAM est une identité IAM que vous pouvez créer dans votre compte et qui dispose d’autorisations spécifiques. Un rôle IAM est similaire à un utilisateur IAM, car il s'agit d'une identité AWS avec des politiques d'autorisation qui déterminent ce que l'identité peut et ne peut pas faire dans AWS. En revanche, au lieu d’être associé de manière unique à une personne, un rôle est conçu pour être assumé par tout utilisateur qui en a besoin. En outre, un rôle ne dispose pas d'informations d'identification standard à long-terme comme un mot de passe ou des clés

d'accès associées. Au lieu de cela, lorsque vous adoptez un rôle, il vous fournit des informations d'identification de sécurité temporaires pour votre session de rôle. Vous pouvez utiliser des rôles pour déléguer l'accès à des utilisateurs, à des applications ou à des services qui n'ont normalement pas accès à vos AWS ressources. Dans ce scénario, vous créez un rôle que Terry Whitlock peut endosser pour accéder à la console de facturation.

Étape 4 : Tester l'accès à la console

Une fois que vous avez terminé les tâches de base, vous êtes prêt à tester la politique. Les tests permettent de s'assurer que la politique fonctionne comme vous le souhaitez. En testant l'accès de chaque utilisateur, vous pouvez comparer les expériences des utilisateurs.

Prérequis

Effectuez les opérations suivantes avant de commencer à suivre les étapes de ce tutoriel :

- Créez un test Compte AWS.
- Connectez-vous à votre test en Compte AWS tant qu'utilisateur root.
- Enregistrez le Compte AWS numéro de votre compte de test afin de pouvoir l'utiliser dans le didacticiel. Par exemple, dans ce tutoriel, nous utilisons le numéro de compte 111122223333. Chaque fois qu'une étape utilise ce numéro de compte, remplacez-le par votre numéro de compte de test.

Étape 1 : Activez l'accès IAM aux informations de facturation sur votre compte de test AWS

Dans ce scénario, vous vous connectez à votre test en Compte AWS tant qu'utilisateur root pour accorder à IAM l'accès aux informations de facturation. Lorsque vous accordez l'accès aux informations de facturation, les utilisateurs et les rôles IAM peuvent accéder à la AWS Billing and Cost Management console. Ce paramètre n'accorde pas aux utilisateurs et aux rôles IAM les autorisations nécessaires pour accéder à ces pages de la console. Il accorde simplement l'accès aux utilisateurs ou aux rôles IAM qui disposent des politiques IAM requises.

> **Note**
>
> Comptes AWS créé à l'aide AWS Organizations de l'activation par défaut de l'accès IAM aux informations de facturation.

Pour activer l'accès de l'utilisateur et du rôle IAM à la Billing and Cost Management console (console de gestion de la facturation et des coûts)

1. Connectez-vous à l' AWS Management Console aide de vos informations d'identification d'utilisateur root (en particulier, l'adresse e-mail et le mot de passe que vous avez utilisés pour créer votre AWS compte).
2. Dans la barre de navigation, sélectionnez le nom de votre compte, puis Compte.
3. Faites défiler la page jusqu'à la section Accès des utilisateurs et des rôles IAM aux données de facturation, puis sélectionnez Modifier.
4. Cochez la case Activate IAM Access (Activer l'accès IAM) pour activer l'accès aux pages de la console de Gestion de la facturation et des coûts.
5. Choisissez Mettre à jour.

   La page affiche le message L'accès des utilisateurs/rôles IAM aux données de facturation est activé.

Dans l'étape suivante de ce tutoriel, vous allez attacher des politiques IAM pour accorder ou refuser l'accès à certaines fonctionnalités de facturation.

Étape 2 : Créer des utilisateurs et des groupes de test

Aucune identité n'est définie pour votre AWS compte de test, à l'exception de l'utilisateur root. Pour permettre l'accès aux informations de facturation, nous créons des identités supplémentaires auxquelles nous pouvons accorder l'autorisation d'accéder aux informations de facturation.

Créer des utilisateurs et des groupes de test

1. Connectez-vous à la console IAM en tant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   > **Note**
   >
   > En tant qu'utilisateur root, vous ne pouvez pas vous connecter à la page Se connecter en tant qu'utilisateur IAM. Si la page Se connecter en tant qu'utilisateur IAM s'affiche, choisissez Se connecter à l'aide de l'adresse e-mail de l'utilisateur root en bas de la page. Pour obtenir de l'aide pour vous connecter en tant qu'utilisateur root, consultez la

section Connexion en AWS Management Console tant qu'utilisateur root dans le Guide de Connexion à AWS l'utilisateur.

2. Dans le volet de navigation, sélectionnez Utilisateurs, puis Ajouter des utilisateurs.

> **Note**
>
> Si IAM Identity Center est activé, un message AWS Management Console s'affiche pour vous rappeler qu'il est préférable de gérer l'accès des utilisateurs dans IAM Identity Center. Dans ce tutoriel, les utilisateurs IAM que nous créons vont découvrir comment fournir un accès aux informations de facturation. Si vous avez créé des utilisateurs dans IAM Identity Center, vous attribuez le jeu d'autorisations de facturation à ces utilisateurs ou groupes à l'aide d'IAM Identity Center au lieu d'IAM.

3. Dans User name (Nom d'utilisateur), saisissez **`pcandella`**. Les noms ne peuvent pas contenir d'espaces.
4. Cochez la case de sélection à côté de Fournir un accès utilisateur au AWS Management Console— facultatif, puis choisissez « Voulez-vous créer un utilisateur IAM ».
5. Sous Mot de passe de la console, sélectionnez Mot de passe généré automatiquement.
6. Décochez la case située en regard de l'option L'utilisateur doit créer un nouveau mot de passe à la prochaine connexion (recommandé), puis sélectionnez Suivant. Comme cet utilisateur IAM est destiné à des fins de test, nous allons télécharger le mot de passe à utiliser lors de la procédure de vérification.
7. Sur la page Définir les autorisations, sous Options d'autorisations, sélectionnez Ajouter un utilisateur au groupe. Ensuite, sous Groupes d'utilisateurs, sélectionnez Créer un groupe.
8. Sur la page Créer un groupe d'utilisateurs, saisissez **`BillingGroup`** dans le champ Nom du groupe d'utilisateurs. Ensuite, sous Politiques d'autorisations, sélectionnez la politique de facturation des fonctions de travail AWS gérées.
9. Sélectionnez Créer un groupe d'utilisateurs pour revenir à la page Définir les autorisations.
10. Sous Groupes d'utilisateurs, cochez la case du **`BillingGroup`** que vous avez créé.
11. Sélectionnez Suivant pour accéder à la page Vérifier et créer.
12. Sur la page Vérifier et créer, vérifiez la liste des appartenances à des groupes d'utilisateurs pour le nouvel utilisateur. Une fois que vous êtes prêt à continuer, sélectionnez Créer un utilisateur.

13. Sur la page Récupérer le mot de passe, sélectionnez Télécharger le fichier .csv pour enregistrer le fichier .csv contenant les informations de connexion de l'utilisateur (URL de connexion, nom d'utilisateur et mot de passe).

    Enregistrez ce fichier pour l'utiliser comme référence lorsque vous vous connectez en AWS tant qu'utilisateur IAM
14. Sélectionnez Retourner à la liste des utilisateurs
15. Répétez cette procédure en apportant les modifications suivantes pour créer l'utilisateur Terry Whitlock et un groupe pour les utilisateurs du support.

    a. À l'étape 3, pour Nom d'utilisateur, saisissez **`twhitlock`**.

    b. À l'étape 8, pour Nom du groupe d'utilisateurs, saisissez **`SupportGroup`**. Ensuite, sous Politiques d'autorisations, sélectionnez la politique relative aux fonctions AWS de tâches gérées. SupportUser

Vous pouvez vérifier les nouveaux utilisateurs, groupes et rôles IAM dans les listes de la console. Pour chaque élément que vous avez créé, vous pouvez sélectionner le nom pour en afficher les détails. Lorsque vous consultez les détails de l'utilisateur, la console affiche la facturation répertoriée sous Politiques d'autorisations pour **`pcandella`** et SupportUserrépertoriée sous Politiques d'autorisations pour**`twhitlock`**.

Pour plus d'informations sur l'utilisation des politiques pour accorder aux utilisateurs IAM l'accès aux fonctionnalités d' AWS Billing and Cost Management , consultez Utilisation de politiques basées sur l'identité (politiques IAM) pour AWS Billing dans le Guide de l'utilisateur AWS Billing .

Étape 3 : Créer un rôle pour accorder l'accès à la console AWS Billing

Vous pouvez utiliser un rôle pour accorder aux utilisateurs IAM l'accès à la console de facturation. Les rôles fournissent des informations d'identification temporaires que les utilisateurs peuvent utiliser en cas de besoin. Dans ce tutoriel, l'utilisateur **`twhitlock`** doit pouvoir accéder aux informations de facturation lorsqu'une demande d'assistance du département financier l'oblige à enquêter sur un problème.

1. Connectez-vous à la console IAM en tant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

> **Note**
>
> En tant qu'utilisateur root, vous ne pouvez pas vous connecter à la page Se connecter en tant qu'utilisateur IAM. Si la page Se connecter en tant qu'utilisateur IAM s'affiche, choisissez Se connecter à l'aide de l'adresse e-mail de l'utilisateur root en bas de la page. Pour obtenir de l'aide pour vous connecter en tant qu'utilisateur root, consultez la section Connexion en AWS Management Console tant qu'utilisateur root dans le Guide de Connexion à AWS l'utilisateur.

2. Dans le panneau de navigation, sélectionnez Utilisateurs, puis sélectionnez l'utilisateur **`twhitlock`** pour en afficher les détails. Copiez l'ARN de l'utilisateur **`twhitlock`** dans le presse-papiers.
3. Dans le panneau de navigation, sélectionnez Rôles, puis Créer un rôle.
4. Sur la page Sélectionner une entité d'approbation, sélectionnez Politique d'approbation personnalisée, puis sous Modifier la déclaration, complétez les éléments suivants :

   - Ajouter des actions pour STS - Vérifiez que cette option AssumeRoleest sélectionnée.
   - Dans Ajouter un principal, sélectionnez Ajouter pour afficher la boîte de dialogue Ajouter un principal. Pour Type de principal, sélectionnez utilisateurs IAM, puis pour ARN, collez l'ARN de l'utilisateur twhitlock que vous avez copié dans le presse-papiers à l'étape 16. Puis, sélectionnez Ajouter le principal.
5. Sélectionnez Suivant pour accéder à la page Ajouter des autorisations.
6. Sous Politiques d'autorisation dans la zone de filtre, entrez **`Billing`** puis sélectionnez la politique de facturation des fonctions AWS de gestion des tâches.
7. Cliquez sur Suivant pour accéder à la page Nommer, vérifier et créer. Sous Nom du rôle, saisissez **`TempBillingAccess`**, puis sélectionnez Créer un rôle.

   Vous recevez une notification indiquant que le rôle a été créé. Affichez le rôle pour consulter les détails le concernant. Dans la section Résumé, prenez note des informations suivantes :

   - Par défaut, la durée maximale de la session d'une heure par défaut. Passé ce délai, l'utilisateur qui a endossé le rôle utilise de nouveau les autorisations de son compte de base. Si l'utilisateur souhaite continuer à utiliser les autorisations du rôle, il doit de nouveau changer de rôle. Vous pouvez modifier le rôle pour augmenter la durée maximale. La durée de session la plus longue possible est de 12 heures.

- Lien pour changer de rôle dans la console. Vous pouvez copier le lien pour le fournir directement aux utilisateurs que vous ajoutez en tant que principaux dans la politique d'approbation. Vous pouvez consulter et modifier la politique d'approbation dans l'onglet Relations d'approbation.

Étape 4 : Tester l'accès à la console

Nous vous recommandons de tester l'accès en vous connectant en tant que chacun des utilisateurs de test afin de vous rendre compte de l'expérience de vos utilisateurs. Suivez les étapes ci-dessous pour vous connecter aux deux comptes de test pour voir les différences de droits d'accès.

Tester l'accès à la facturation en se connectant avec les deux utilisateurs de test

1. Utilisez votre AWS identifiant ou alias de compte, votre nom d'utilisateur IAM et votre mot de passe pour vous connecter à la console IAM.

   > **Note**
   >
   > Pour votre commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser votre nom d'utilisateur IAM et les informations de votre compte. Si vous vous êtes déjà connecté en tant qu'utilisateur différent, sélectionnez Sign in to a different account (Se connecter à un compte différent) en bas de la page pour revenir à la page de connexion principale. À partir de là, vous pouvez saisir votre identifiant de AWS compte ou votre alias de compte pour être redirigé vers la page de connexion utilisateur IAM de votre compte.

2. Connectez-vous avec chaque utilisateur en suivant la procédure fournie ci-dessous afin de pouvoir comparer les différentes expériences utilisateur.

   Accès complet

   a. Connectez-vous à votre compte Compte AWS en tant qu'utilisateur**`pcandella`**.

   b. Dans la barre de navigation, choisissez pcandella@111122223333, puis sélectionnez Tableau de bord de facturation.

   c. Parcourez les pages et sélectionnez les différents boutons pour vous assurer de disposer de toutes les autorisations de modification.

Pas d'accès

a. Connectez-vous à votre compte Compte AWS en tant qu'utilisateur**`twhitlock`**.
b. Dans la barre de navigation, choisissez twhitlock@111122223333, puis sélectionnez Tableau de bord de facturation.
c. Un message s'affiche indiquant que vous avez besoin d'autorisations. Aucune donnée de facturation n'est visible.

Changer de rôle pour améliorer l'accès

a. Connectez-vous à votre compte Compte AWS en tant qu'utilisateur**`twhitlock`**.
b. Dans la barre de navigation, choisissez twhitlock@111122223333, puis sélectionnez Changer de rôle.

   La page Changer de rôle s'ouvre. Complétez les informations comme suit :

   - Account-111122223333
   - Role-**`TempBillingAccess`**

   Sélectionnez Changer de rôle

   Vous pouvez également utiliser l'URL fournie dans Lien pour changer de rôle dans la console afin d'ouvrir la page Changer de rôle.
c. La console affiche le AWS Billing tableau de bord et la barre de navigation affiche TempBillingAccess@111122223333.

Récapitulatif

Vous avez maintenant terminé les étapes nécessaires pour permettre aux utilisateurs IAM d'accéder à la console AWS Billing . Par conséquent, vous avez vu de première main ce qu'est l'expérience de la console de facturation de vos utilisateurs. Vous pouvez maintenant procéder à l'implémentation de cette logique dans votre environnement de production à votre convenance.

Ressources connexes

Pour obtenir des informations connexes contenues dans le Guide de l'utilisateur AWS Billing , veuillez consulter les ressources suivantes :

- Activation de l'accès à la AWS Billing console
- AWS Exemples de politiques de facturation
- Utilisation de politiques basées sur l'identité (politiques IAM) pour la facturation AWS
- Migration du contrôle d'accès pour AWS Billing

Pour obtenir des informations connexes contenues dans le IAM Guide de l'utilisateur, veuillez consulter les ressources suivantes :

- Politiques gérées et politiques en ligne
- Contrôler l'accès des utilisateurs IAM à l' AWS Management Console
- Attacher une politique à un groupe IAM

Didacticiel IAM : déléguer l'accès entre des comptes AWS à l'aide des rôles IAM

Ce tutoriel vous explique comment utiliser un rôle pour déléguer l'accès aux ressources situées dans différents Comptes AWS vous appartenant, un rôle appelé Production et Development (Développement). Vous partagez les ressources d'un compte avec les utilisateurs d'un autre compte. En configurant l'accès entre les comptes de cette manière, vous n'avez pas besoin de créer des utilisateurs IAM individuels dans chaque compte. En outre, les utilisateurs n'ont pas besoin de se déconnecter d'un compte et de se connecter à un autre compte pour accéder à des ressources situées dans des Comptes AWS différents. Après avoir configuré le rôle, vous découvrirez comment utiliser le rôle à partir du AWS Management Console AWS CLI, et de l'API.

> **Note**
>
> Les politiques IAM basées sur les ressources et les rôles ne délèguent l'accès entre les comptes qu'au sein d'une seule partition. Par exemple, supposons que vous avez un compte dans la région USA Ouest (Californie du Nord) sur la partition `aws` standard. Vous avez également un compte dans la région Chine (Beijing) sur la partition `aws-cn`. Vous ne pouvez pas utiliser une politique basée sur les ressources d'Amazon S3 dans votre compte en Chine (Beijing) pour autoriser l'accès aux utilisateurs de votre compte `aws` standard.

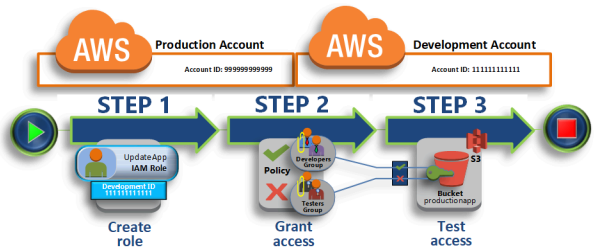
Dans ce tutoriel, le compte production gère les applications en direct. Les développeurs et les testeurs utilisent le compte développement comme environnement de test (sandbox) pour tester librement les applications. Dans chaque compte, vous stockez les informations de l'application dans des compartiments Amazon S3. Vous gérez des utilisateurs IAM dans le compte développement, où vous disposez de deux groupes d'utilisateurs IAM : les développeurs et les testeurs. Les utilisateurs des deux groupes d'utilisateurs ont des autorisations pour travailler dans le compte Développement et y accéder aux ressources qui s'y trouvent. De temps à autre, un développeur doit mettre à jour les applications en direct dans le compte production. Les développeurs stockent ces applications dans un compartiment Amazon S3 appelé `productionapp`.

Au terme de ce tutoriel, vous disposerez des éléments suivants :

- Utilisateurs du compte développement (le compte approuvé) autorisés à endosser un rôle spécifique dans le compte production.
- Rôle du compte production (le compte d'approbation) autorisé à accéder à un compartiment Amazon S3 spécifique.
- Le compartiment `productionapp` dans le compte production.

Les développeurs peuvent utiliser le rôle indiqué dans le AWS Management Console pour accéder au `productionapp` compartiment dans le compte Production. Ils peuvent également accéder au compartiment à l'aide des appels API authentifiés par les informations d'identification temporaires fournies par le rôle. Les tentatives similaires des testeurs pour utiliser le rôle échouent.

Ce flux de travail se compose de trois étapes de base :



Créer un rôle dans le compte production

Tout d'abord, vous utilisez le AWS Management Console pour établir un lien de confiance entre le compte de production (numéro d'identification 999999999999) et le compte de développement (numéro d'identification 111111111111). Vous commencez par créer un rôle IAM nommé UpdateApp. Lorsque vous créez le rôle, vous définissez le compte développement en tant

qu'entité approuvée et spécifiez une politique d'autorisations qui autorise les utilisateurs de confiance à mettre à jour le compartiment `productionapp`.

[Accorder l'accès au rôle]

Dans cette section, vous modifiez la politique de groupe d'utilisateurs IAM pour refuser aux testeurs l'accès au rôle `UpdateApp`. Parce que les testeurs PowerUser y ont accès dans ce scénario, vous devez explicitement refuser la possibilité d'utiliser le rôle.

[Tester l'accès en changeant de rôles]

Enfin, en tant que développeur, vous utilisez le rôle `UpdateApp` pour mettre à jour le compartiment `productionapp` dans le compte production. Vous allez voir comment accéder au rôle via la AWS console, le AWS CLI, et l'API.

Prérequis

Le didacticiel présume que vous avez déjà ce qui suit en place :

- Deux éléments distincts Comptes AWS que vous pouvez utiliser, l'un pour représenter le compte de développement et l'autre pour le compte de production.
- Les utilisateurs et les groupes d'utilisateurs du compte développement créés et configurés comme suit :

Utilisateur	Groupe d'utilisateurs	Autorisations
David	Développeurs	Les deux utilisateurs peuvent se connecter et utiliser AWS Management Console le compte de développement.
Jane	Testeurs	

- Vous n'avez pas besoin d'avoir des utilisateurs ou des groupes d'utilisateurs créés dans le compte production.
- Un compartiment Amazon S3 créé dans le compte production. Vous pouvez l'appeler `ProductionApp` dans ce tutoriel, mais du fait que les noms de compartiment S3 doivent être globalement uniques, vous devrez utiliser un compartiment avec un nom différent.

Créer un rôle dans le compte production

Vous pouvez autoriser les utilisateurs de l'un Compte AWS à accéder aux ressources d'un autre Compte AWS. Pour ce faire, créez un rôle qui définit qui peut y accéder et quelles autorisations il accorde aux utilisateurs qui y basculent.

Dans cette étape du tutoriel, vous créez le rôle dans le compte production et spécifiez le compte développement comme entité approuvée. Vous limitez également les autorisations du rôle à un accès en lecture seule et en écriture au compartiment `productionapp`. Toute personne ayant autorisation d'utiliser le rôle peut lire et écrire dans le compartiment `productionapp`.

Avant de créer un rôle, vous avez besoin de l'ID de compte du développement Compte AWS. Chacun Compte AWS possède un identifiant de compte unique qui lui est attribué.

Pour obtenir l' Compte AWS ID de développement

1. Connectez-vous au compte de développement en AWS Management Console tant qu'administrateur et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans la barre de navigation, sélectionnez Support, puis Support Center (Centre de support). Le numéro de compte (ID) à 12 chiffres avec lequel vous êtes actuellement connecté apparaît dans le panneau de navigation Support Center (Centre de support). Pour ce scénario, vous pouvez utiliser l'ID 111111111111 pour le compte développement. Toutefois, vous devez utiliser un ID de compte valide si vous utilisez ce scénario dans votre environnement de test.

Pour créer un rôle dans le compte de production qui peut être utilisé par le compte de développement

1. Connectez-vous en AWS Management Console tant qu'administrateur du compte de production et ouvrez la console IAM.
2. Avant de créer le rôle, préparez la politique gérée qui définit les autorisations pour les exigences du rôle. Vous attachez cette politique au rôle dans une étape ultérieure.

   Vous devez définir l'accès en lecture et en écriture au compartiment `productionapp`. Bien que AWS certaines politiques soient gérées par Amazon S3, aucune ne fournit un accès en lecture et en écriture à un seul compartiment Amazon S3. Vous pouvez créer votre propre politique à la place.

   Dans le panneau de navigation, choisissez Politiques, puis Créer une politique.

3. Choisissez l'onglet JSON et copiez le texte du document de politique JSON suivant. Collez ce texte dans la zone de texte JSON, en remplaçant l'ARN de la ressource (`arn:aws:s3:::productionapp`) par celui qui correspond véritablement à votre compartiment Amazon S3.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation"
       ],
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

   L'action `ListAllMyBuckets` accorde l'autorisation de répertorier tous les compartiments appartenant à l'expéditeur authentifié de la demande. L'autorisation `ListBucket` permet aux utilisateurs d'afficher des objets dans le compartiment `productionapp`. Les autorisations `GetObject`, `PutObject`, `DeleteObject` permettent aux utilisateurs d'afficher, de mettre à jour et de supprimer le contenu du compartiment `productionapp`.

4. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez Restructuration de politique.

5. Sur la page Vérifier et créer, tapez **`read-write-app-bucket`** pour le nom de la politique. Vérifiez les autorisations accordées par votre politique, puis choisissez Créer une politique pour enregistrer votre travail.

   La nouvelle politique apparaît dans la liste des politiques gérées.

6. Dans le panneau de navigation, choisissez Roles (Rôles), puis Create role (Créer un rôle).
7. Choisissez le type de rôle Un Compte AWS.
8. Pour l'ID de compte, saisissez l'ID du compte développement.

   Ce tutoriel utilise l'exemple d'ID de compte **`111111111111`** pour le compte développement. Vous devez utiliser un ID de compte valide. Si vous utilisez un ID de compte non valide, comme **`111111111111`**, IAM ne vous laisse pas créer de rôle.

   Pour le moment, les utilisateurs n'ont pas besoin d'un ID externe ou d'une authentification multi-facteurs (MFA) pour endosser le rôle. Laissez ces options décochées. Pour plus d'informations, veuillez consulter Utilisation de l'authentification multifactorielle (MFA) dans AWS.

9. Choisissez suivant : autorisations pour configurer les autorisations associées au rôle.
10. Cochez la case en regard de la politique que vous avez créée précédemment.

> **Conseil**
>
> Pour filtre, sélectionnez politiques gérées par le client pour affiner la liste afin d'inclure uniquement les politiques que vous avez créées. Cela masque les politiques créées par AWS et permet de rechercher plus facilement celle dont vous avez besoin.

Ensuite, choisissez Suivant.

11. (Facultatif) Ajoutez des métadonnées au rôle en associant les identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Balisage des ressources IAM.
12. (Facultatif) Pour Description, saisissez une description pour le nouveau rôle.
13. Après avoir procédé à la révision du rôle, sélectionnez Créer un rôle.

    Le rôle `UpdateApp` s'affiche dans la liste des rôles.

À présent, vous devez obtenir l'Amazon Resource Name (ARN) du rôle, qui est un identifiant unique du rôle. Lorsque vous modifiez la politique de groupe d'utilisateurs développeurs et testeurs, vous spécifiez l'ARN du rôle pour accorder ou refuser les autorisations.

Pour obtenir l'ARN pour UpdateApp

1. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles).
2. Dans la liste des rôles, sélectionnez le rôle `UpdateApp`.
3. Dans la section Récapitulatif du panneau des détails, copiez la valeur du champ ARN de rôle.

   Le compte Production ayant pour ID de compte 999999999999, l'ARN du rôle est donc `arn:aws:iam::999999999999:role/UpdateApp`. Assurez-vous de fournir le véritable Compte AWS identifiant du compte de production.

À ce stade, vous avez établi la confiance entre les comptes production et développement. Pour ce faire, créez un rôle dans le compte production qui identifie le compte développement en tant que principal compte de confiance. Vous avez également défini ce que les utilisateurs qui basculent vers le rôle `UpdateApp` peuvent faire.

Ensuite, modifiez les autorisations pour les groupes d'utilisateurs.

Accorder l'accès au rôle

À ce stade, les membres des groupes d'utilisateurs testeurs et développeurs disposent d'autorisations leur permettant de tester librement des applications dans le compte développement. Utilisez la procédure requise pour ajouter des autorisations visant à autoriser le changement de rôle.

Pour modifier le groupe d'utilisateurs des développeurs afin de leur permettre de passer au UpdateApp rôle

1. Connectez-vous en tant qu'administrateur dans le compte développement, puis ouvrez la console IAM.
2. Sélectionnez User groups (Groupes d'utilisateurs), puis Developers (Développeurs).
3. Sélectionnez l'onglet Permissions (Autorisations), puis Add permissions (Ajouter des autorisations), et enfin Create inline policy (Créer une politique en ligne).
4. Sélectionnez l'onglet JSON.
5. Ajoutez l'instruction de politique suivante pour autoriser l'action `AssumeRole` sur le rôle `UpdateApp` dans le compte Production. Assurez-vous de remplacer *`PRODUCTION-ACCOUNT-ID`* dans l'`Resource`élément par l' Compte AWS ID réel du compte de production.

   ```
   {
     "Version": "2012-10-17",
     "Statement": {
       "Effect": "Allow",
       "Action": "sts:AssumeRole",
       "Resource": "arn:aws:iam::PRODUCTION-ACCOUNT-ID:role/UpdateApp"
     }
   }
   ```

   L'effet `Allow` autorise explicitement l'accès du groupe Développeurs au rôle `UpdateApp` dans le compte Production. Tout développeur qui essaie d'accéder à ce rôle y parvient.
6. Choisissez Examiner une politique.
7. Tapez un Nom, tel que **`allow-assume-S3-role-in-production`**.
8. Choisissez Créer une politique.

Dans la plupart des environnements, vous n'aurez peut-être pas besoin de la procédure suivante. Toutefois, si vous utilisez PowerUserAccess des autorisations, il est possible que certains groupes soient déjà en mesure de changer de rôle. Les procédures suivantes indiquent comment ajouter une autorisation `"Deny"` au groupe Testeurs pour s'assurer qu'ils ne peuvent pas endosser le rôle. Si vous n'avez pas besoin de cette procédure dans votre environnement, nous vous recommandons de ne pas l'ajouter. Les autorisations `"Deny"` rendent l'ensemble des autorisations plus compliqué à gérer et à comprendre. Utilisez les autorisations `"Deny"` uniquement lorsque vous ne disposez pas de meilleure option.

Pour modifier le groupe d'utilisateurs testeurs afin de lui refuser l'autorisation d'endosser le rôle **`UpdateApp`**

1. Choisissez User groups (groupes d'utilisateurs), puis Testers (testeurs).
2. Sélectionnez l'onglet Permissions (Autorisations), puis Add permissions (Ajouter des autorisations), et enfin Create inline policy (Créer une politique en ligne).
3. Sélectionnez l'onglet JSON.
4. Ajoutez l'instruction de politique suivante pour refuser l'action `AssumeRole` sur le rôle `UpdateApp`. Assurez-vous de remplacer *`PRODUCTION-ACCOUNT-ID`* dans l'`Resource`élément par l' Compte AWS ID réel du compte de production.

   ```
   {
     "Version": "2012-10-17",
     "Statement": {
       "Effect": "Deny",
       "Action": "sts:AssumeRole",
       "Resource": "arn:aws:iam::PRODUCTION-ACCOUNT-ID:role/UpdateApp"
     }
   }
   ```

   L'effet `Deny` refuse explicitement l'accès du groupe Testeurs au rôle `UpdateApp` dans le compte Production. Tout testeur qui essaiera d'accéder au rôle recevra un message d'accès refusé.
5. Choisissez Examiner une politique.
6. Tapez un Nom comme **`deny-assume-S3-role-in-production`**.
7. Sélectionnez Créer une politique.

Le groupe d'utilisateurs Développeurs a maintenant l'autorisation d'utiliser le rôle `UpdateApp` dans le compte Production. Le groupe d'utilisateurs Testeurs ne peut pas utiliser le rôle `UpdateApp`.

Ensuite, vous pouvez voir comment David, un développeur, peut accéder au compartiment `productionapp` dans le compte production. David peut accéder au bucket depuis le AWS Management Console AWS CLI, le ou l' AWS API.

Tester l'accès en changeant de rôles

Après avoir terminé les deux premières étapes de ce tutoriel, vous disposez d'un rôle qui accorde l'accès à une ressource du compte production. Vous disposez également d'un groupe d'utilisateurs

dans le compte développement avec des utilisateurs autorisés à utiliser ce rôle. Cette étape explique comment tester le passage à ce rôle depuis l'API AWS Management Console AWS CLI, le et l' AWS API.

> **Important**
>
> Il est possible de passer à un rôle uniquement si vous vous connectez en tant qu'utilisateur IAM ou en tant qu'utilisateur fédéré. En outre, si vous lancez une instance Amazon EC2 pour exécuter une application, l'application peut endosser un rôle via son profil d'instance. Il n'est pas possible de passer à un rôle si vous vous connectez en tant qu' Utilisateur racine d'un compte AWS.

Changer de rôle (console)

Si David doit travailler dans l'environnement de production du AWS Management Console, il peut le faire en utilisant Switch Role. Il indique l'ID de compte ou l'alias et le nom du rôle, et ses autorisations passent immédiatement à celles autorisées par le rôle. Il peut ensuite utiliser la console pour travailler avec le compartiment `productionapp`, mais il ne peut pas utiliser d'autres ressources de l'environnement production. Bien que David utilise le rôle, il ne peut pas non plus utiliser ses privilèges d'utilisateur avancé dans le compte développement. Ceci est dû au fait qu'un seul ensemble d'autorisations peut être actif à la fois.

> **Important**
>
> Le changement de rôle à l'aide du AWS Management Console ne fonctionne qu'avec les comptes qui ne nécessitent pas de`ExternalId`. Par exemple, supposons que vous accordiez l'accès à votre compte à un tiers et que vous ayez besoin d'un `ExternalId` dans un élément `Condition` de votre politique d'autorisations. Dans ce cas, le tiers ne peut accéder à votre compte qu'à l'aide de l' AWS API ou d'un outil de ligne de commande. Le tiers ne peut pas utiliser la console, car elle ne peut pas fournir de valeur pour `ExternalId`. Pour plus d'informations sur ce scénarioComment utiliser un identifiant externe lorsque vous accordez l'accès à vos AWS ressources à un tiers, consultez la section « Comment activer l'accès entre comptes » AWS Management Console dans le blog sur la AWS sécurité.

IAM fournit deux procédures que David peut utiliser pour accéder à la page Switch Role (changer de rôle) :

- David reçoit un lien de son administrateur qui dirige vers la configuration Switch Role prédéfinie. Le lien est fourni à l'administrateur sur la page finale de l'assistant Créer un rôle ou sur la page Résumé du rôle pour un rôle inter-compte. Si David choisit ce lien, il accède à la page Switch Role (Changer de rôle) avec les champs ID de compte et Nom du rôle déjà remplis. Il ne reste plus à David qu'à choisir Switch Role (Changer de rôle).
- L'administrateur n'envoie pas le lien dans un e-mail, mais il envoie les valeurs de l'ID de compte et du Nom de rôle. Pour changer de rôle, David doit manuellement saisir les valeurs. La procédure suivante en est l'illustration.

Pour endosser un rôle

1. David se connecte en AWS Management Console utilisant son utilisateur normal dans le groupe d'utilisateurs de développement.
2. Il choisit le lien que l'administrateur lui a envoyé par email. Cela amène David à la page Switch Role (Changer de rôle) avec l'identifiant ou l'alias de compte et les informations sur le nom du rôle déjà remplis.

   —ou—

   David choisit son nom (le menu Identity [Identité]) dans la barre de navigation, puis sélectionne Switch Roles (Changer de rôle).

   Si c'est la première fois que David essaie d'accéder à la page Switch Role (changer de rôle) de cette manière, il est dirigé vers la 1ère page de mise en route Switch Role (changer de rôle). Cette page fournit des informations supplémentaires sur la manière dont le changement de rôle permet aux utilisateurs de gérer des ressources entre Comptes AWS. David doit choisir Switch Role (changer de rôle) sur cette page pour appliquer le reste de la procédure.
3. Ensuite, pour accéder au rôle, David doit saisir manuellement le numéro d'ID (`999999999999`) et le nom du rôle (`UpdateApp`) du compte Production.

   En outre, David souhaite contrôler les rôles et les autorisations associées actuellement actifs dans IAM. Pour suivre ces informations, il saisit `PRODUCTION` dans la zone de texte Display Name (nom complet), choisit l'option de couleur rouge, puis choisit Switch Role (changer de rôle).
4. David peut maintenant utiliser la console Amazon S3 pour travailler avec le compartiment Amazon S3 ou toute autre ressource pour laquelle le rôle `UpdateApp` dispose d'autorisations.

5. Quand c'est fait, David peut retourner dans ses autorisations d'origine. Pour cela, il choisit le nom complet du rôle PRODUCTION sur la barre de navigation, puis il sélectionne Back to David @ 111111111111 (Revenir à David @ 111111111111).
6. La prochaine fois que David voudra changer de rôle et choisira le menu Identity (identité) dans la barre de navigation, il verra l'entrée PRODUCTION toujours affichée depuis la dernière fois. Il lui suffira de choisir cette entrée pour changer de rôle immédiatement sans avoir à ressaisir l'ID du compte et le nom du rôle.

Changer de rôles (AWS CLI)

Si David a besoin de travailler dans l'environnement production dans la ligne de commande, il peut y parvenir grâce à l'outil AWS CLI. Il exécute la commande `aws sts assume-role` et transmet l'ARN du rôle pour obtenir les informations d'identification de sécurité temporaires de ce rôle. Il configure ensuite ces informations d'identification dans les variables d'environnement afin que AWS CLI les commandes suivantes fonctionnent en utilisant les autorisations du rôle. Bien que David utilise le rôle, il ne peut pas utiliser ses privilèges d'utilisateur avancé dans le compte développement, car un seul ensemble d'autorisations peut être effectif à la fois.

Notez que toutes les clés d'accès et tous les jetons sont des exemples uniquement et ne peuvent pas être utilisés comme indiqué. Remplacez-les par les valeurs correspondantes de votre environnement en direct.

Pour endosser un rôle

1. David ouvre une fenêtre d'invite de commande et confirme que le AWS CLI client fonctionne en exécutant la commande :

   ```
   aws help
   ```

   > **Note**
   >
   > L'environnement par défaut de David utilise les informations d'identification de l'utilisateur `David` de son profil par défaut qu'il a créé avec la commande `aws configure`. Pour plus d'informations, veuillez consulter configuration de l'outil AWS Command Line Interface dans le guide de l'utilisateur de l'outil AWS Command Line Interface .

2. Il commence le processus de changement de rôle en exécutant la commande suivante pour passer au rôle `UpdateApp` dans le compte production. Il a reçu l'ARN du rôle auprès de

l'administrateur ayant créé le rôle. La commande nécessite que vous fournissiez un nom de session également. Pour cela, vous pouvez choisir n'importe quel texte.

```
aws sts assume-role --role-arn "arn:aws:iam::999999999999:role/UpdateApp" --role-
session-name "David-ProdUpdate"
```

David voit ensuite ce qui suit dans le résultat :

```
{
    "Credentials": {
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": "AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLE
CvSRyh0FW7jEXAMPLEW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
uZEXAMPLECihzFB5lTYLto9dyBgSDy
EXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPyOj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3Uuysg
sKdEXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP
+4eZScEXAMPLEsnf87e
NhyDHq6ikBQ==",
        "Expiration": "2014-12-11T23:08:07Z",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

3. David voit les trois éléments dont il a besoin dans la section informations d'identification du résultat.

   - `AccessKeyId`
   - `SecretAccessKey`
   - `SessionToken`

   David doit configurer l' AWS CLI environnement pour utiliser ces paramètres lors des appels suivants. Pour plus d'informations sur les différentes manières de configurer vos informations d'identification, veuillez consulter Configuration de l'outil AWS Command Line Interface. Vous ne pouvez pas utiliser la commande `aws configure`, car elle ne prend pas en charge la capture du jeton de session. En revanche, vous pouvez saisir manuellement les informations dans un fichier de configuration. Du fait qu'il s'agisse d'informations d'identification temporaires avec un

délai d'expiration relativement court, il est plus facile de les ajouter à l'environnement de votre session de ligne de commande actuelle.

4. Pour ajouter les trois valeurs à l'environnement, David coupe et colle le résultat de l'étape précédente dans les commandes suivantes. Vous pourriez vouloir couper et coller dans un simple éditeur de texte pour résoudre les problèmes de retour à la ligne dans le résultat du jeton de session. Elles doivent être ajoutées sous la forme d'une simple chaîne longue, même si elles s'affichent avec des retours de ligne pour plus de clarté.

> **Note**
>
> L'exemple suivant illustre les commandes fournies dans l'environnement Windows où « set » est la commande destinée à créer une variable d'environnement. Sur Linux ou macOS, vous devez plutôt utiliser la commande « export ». Toutes les autres sections de l'exemple sont valides dans les trois environnements.
> Pour en savoir plus sur l'utilisation des outils pour Windows Powershell, consultez Passer à un rôle IAM (Outils pour Windows PowerShell)

```
set AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
set AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
set AWS_SESSION_TOKEN=AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvS
Ryh0FW7jEXAMPLEW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
uZEXAMPLECihzFB5lTYLto9dyBgSDyEXA
MPLEKEY9/
g7QRUhZp4bqbEXAMPLENwGPyOj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UusKd
EXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP
+4eZScEXAMPLENhykxiHen
DHq6ikBQ==
```

À ce stade, toutes les commandes seront exécutées avec les autorisations du rôle identifié par ces informations d'identification. Dans le cas de David, le rôle `UpdateApp`.

5. Exécutez la commande pour accéder aux ressources du compte Production. Dans cet exemple, David répertorie simplement le contenu de son compartiment S3 avec la commande suivante.

```
aws s3 ls s3://productionapp
```

Du fait que les noms de compartiments Amazon S3 sont universellement uniques, il n'est pas nécessaire de spécifier l'ID du compte qui est titulaire du compartiment. Pour accéder aux ressources d'autres AWS services, reportez-vous à la AWS CLI documentation de ce service pour connaître les commandes et la syntaxe requises pour référencer ses ressources.

Utilisation AssumeRole (AWS API)

Lorsque David a besoin de faire une mise à jour dans le compte production depuis le code, il réalise un appel `AssumeRole` pour endosser le rôle `UpdateApp`. L'appel renvoie des informations d'identification temporaires qu'il peut utiliser pour accéder au compartiment `productionapp` dans le compte production. Grâce à ces informations d'identification, David peut réaliser des appels d'API pour mettre à jour le compartiment `productionapp`. Cependant, il ne peut pas réaliser des appels d'API pour accéder aux autres ressources du compte production, même s'il a des autorisations d'utilisateur avancé dans le compte développement.

Pour endosser un rôle

1. David appelle `AssumeRole` dans le cadre d'une application. Il doit spécifier l'ARN `UpdateApp` : `arn:aws:iam::999999999999:role/UpdateApp`.

   La réponse de l'appel `AssumeRole` inclut les informations d'identification temporaires avec un `AccessKeyId` et un `SecretAccessKey`. Elle inclut également une heure `Expiration` qui indique à quel moment les informations d'identification expirent et vous devez en demander de nouvelles.
2. Grâce aux informations d'identification de sécurité temporaires, David réalise un appel `s3:PutObject` pour mettre à jour le compartiment `productionapp`. Il transfère les informations d'identification à l'appel d'API en tant que paramètre `AuthParams`. Du fait que les informations d'identification de rôle temporaires ont uniquement accès en lecture et en écriture au compartiment `productionapp`, toute autre action dans le compte Production est refusée.

Pour obtenir un exemple de code (à l'aide de Python), veuillez consulter Passage à un rôle IAM (AWS API).

Ressources connexes

- Pour plus d'informations sur les utilisateurs et groupes d'utilisateurs IAM, veuillez consulter Identités IAM (utilisateurs, groupes d'utilisateurs et rôles).

- Pour plus d'informations sur les compartiments Amazon S3, veuillez consulter créer un compartiment dans le Amazon Simple Storage Service User Guide (guide de l'utilisateur du service de stockage simple de Amazon).
- Pour savoir si les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, veuillez consulter Qu'est-ce que l'Analyseur d'accès IAM ?.

Récapitulatif

Vous avez terminé le didacticiel d'accès aux API entre les comptes. Vous avez créé un rôle pour établir des relations de confiance avec un autre compte et défini les actions que les entités approuvées peuvent effectuer. Ensuite, vous avez modifié une politique de groupe d'utilisateurs pour contrôler les utilisateurs IAM qui peuvent accéder au rôle. Par conséquent, les développeurs du compte développement peuvent faire des mises à jour dans le compartiment `productionapp` du compte production à l'aide d'informations d'identification temporaires.

Didacticiel IAM : créer et attacher votre première politique gérée par le client

Dans ce didacticiel, vous allez utiliser le AWS Management Console pour créer une politique gérée par le client, puis associer cette politique à un utilisateur IAM de votre Compte AWS. La politique que vous créez permet à un utilisateur de test IAM de se connecter directement au AWS Management Console avec des autorisations en lecture seule.

Ce flux de travail se compose de trois étapes de base :

Étape 1 : Créer la politique

Par défaut, les utilisateurs IAM ne sont autorisés à rien faire. Ils ne peuvent pas accéder à la Console de gestion AWS ou à gérer les données qu'elle contient sans votre autorisation. Dans cette étape, vous créez une politique gérée par le client qui autorise tous les utilisateurs attachés à se connecter à la console.

Étape 2 : Attacher la politique

Lorsque vous attachez une politique à un utilisateur, celui-ci hérite des autorisations d'accès associées à cette politique. Dans cette étape, vous attachez la nouvelle politique à un utilisateur test.

[Étape 3 : Tester l'accès utilisateur](#)

Une fois la politique attachée, vous pouvez vous connecter en tant que l'utilisateur pour tester la politique.

Prérequis

Pour exécuter les étapes de ce didacticiel, vous devez déjà disposer des éléments suivants :

- Et Compte AWS auquel vous pouvez vous connecter en tant qu'utilisateur IAM avec des autorisations administratives.
- Un utilisateur IAM de test n'a aucune autorisation attribuée et n'appartient à aucun groupe comme suit :

Nom utilisateur	Groupe	Autorisations
PolicyUser	<aucune>	<aucune>

Étape 1 : Créer la politique

Au cours de cette étape, vous créez une politique gérée par le client qui permet à tout utilisateur attaché de se connecter AWS Management Console avec un accès en lecture seule aux données IAM.

Pour créer la politique de votre utilisateur de test

1. Connectez-vous à la console IAM à l'adresse https://console.aws.amazon.com/iam/ en tant qu'utilisateur disposant d'autorisations Administrateur.
2. Dans le panneau de navigation, sélectionnez Policies (Politiques).
3. Dans le panneau de contenu, sélectionnez Créer une politique.
4. Choisissez l'option JSON et copiez le texte du document de politique JSON suivant. Collez ce texte dans la zone de texte JSON.

```
{
    "Version": "2012-10-17",
    "Statement": [ {
        "Effect": "Allow",
```

```
        "Action": [
            "iam:GenerateCredentialReport",
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    } ]
}
```

5. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou sélectionnez Examiner une politique dans l'onglet Editeur visuel, IAM peut restructurer votre politique pour optimiser son affichage dans l'éditeur visuel. Pour plus d'informations, consultez Restructuration de politique.

6. Sur la page Vérifier et créer, tapez **`UsersReadOnlyAccessToIAMConsole`** pour le nom de la politique. Vérifiez les autorisations accordées par votre politique, puis choisissez Créer une politique pour enregistrer votre travail.

   La nouvelle politique s'affiche dans la liste des politiques gérées et est prête à être attachée.

Étape 2 : Attacher la politique

Ensuite, vous attachez la politique que vous venez de créer à votre utilisateur IAM de test.

Pour attacher la politique à votre utilisateur de test

1. Dans le panneau de navigation de la console IAM, sélectionnez Policies (Politiques).
2. En haut de la liste des politiques, dans la zone de recherche, commencez à taper **`UsersReadOnlyAccesstoIAMConsole`** jusqu'à ce que vous puissiez voir votre politique. Choisissez ensuite le bouton radio situé à côté de UsersReadOnlyAccessToIAMConsole dans la liste.
3. Cliquez sur le bouton Actions, puis choisissez Attach (Attacher).
4. Dans les entités IAM, choisissez l'option de filtrage pour les Utilisateurs.

5. Dans la zone de recherche, commencez à taper **`PolicyUser`** jusqu'à ce que cet utilisateur soit visible dans la liste. Cochez ensuite la case en regard de cet utilisateur dans la liste.
6. Choisissez Attach policy (Attacher une politique).

Vous avez attaché la politique à votre utilisateur de test IAM, ce qui signifie que l'utilisateur dispose maintenant d'un accès en lecture seule à la console IAM.

Étape 3 : Tester l'accès utilisateur

Dans le cadre de ce didacticiel, nous vous recommandons de tester l'accès en vous connectant en tant que chacun des utilisateurs de test afin de vous rendre compte de l'expérience que vivent vos utilisateurs.

Tester l'accès en vous connectant avec votre utilisateur test

1. Connectez-vous à la console IAM à l'adresse https://console.aws.amazon.com/iam/ en tant qu'utilisateur test `PolicyUser`.
2. Parcourez les pages de la console et essayez de créer un nouvel utilisateur ou un nouveau groupe. Notez que `PolicyUser` peut afficher des données mais ne peut ni créer, ni modifier des données IAM existantes.

Ressources connexes

Pour plus d'informations, consultez les ressources suivantes :

- Politiques gérées et politiques en ligne
- Contrôler l'accès des utilisateurs IAM à l' AWS Management Console

Récapitulatif

Vous avez terminé toutes les étapes nécessaires pour créer et attacher une politique gérée par l'utilisateur. Par conséquent, vous pouvez vous connecter à la console IAM avec votre compte de test pour voir à quoi ressemble l'expérience de vos utilisateurs.

Tutoriel IAM : définir les autorisations d'accès aux AWS ressources en fonction des balises

Le contrôle d'accès basé sur les attributs (ABAC) est une stratégie d'autorisation qui définit des autorisations en fonction des attributs. Dans AWS, ces attributs sont appelés balises. Vous pouvez associer des balises aux ressources IAM, notamment aux entités IAM (utilisateurs ou rôles) et aux AWS ressources. Vous pouvez définir des politiques qui utilisent des clés de condition de balise pour accorder des autorisations à vos principaux en fonction de leurs balises. Lorsque vous utilisez des balises pour contrôler l'accès à vos AWS ressources, vous permettez à vos équipes et à vos ressources de se développer en modifiant moins les AWS politiques. Les politiques ABAC sont plus flexibles que AWS les politiques traditionnelles, qui vous obligent à répertorier chaque ressource individuelle. Pour de plus amples informations sur l'ABAC et ses avantages par rapport aux politiques traditionnelles, veuillez consulter À quoi sert ABAC ? AWS.

> **Note**
>
> Vous devez transmettre une valeur unique pour chaque balise de session. AWS Security Token Service ne prend pas en charge les balises de session à valeurs multiples.

Rubriques

- Présentation du didacticiel
- Prérequis
- Étape 1 : Créer des utilisateurs test
- Étape 2 : Créer la politique ABAC
- Étape 3 : Créer les rôles
- Étape 4 : Tester la création de secrets
- Étape 5 : Tester l'affichage des secrets
- Étape 6 : Tester l'adaptabilité
- Étape 7 : Tester la mise à jour et la suppression des secrets
- Récapitulatif
- Ressources connexes
- Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC

Présentation du didacticiel

Ce didacticiel explique comment créer et tester une politique qui autorise des rôles IAM avec des balises de principal à accéder aux ressources dont les balises correspondent. Lorsqu'un principal fait une demande à AWS, les autorisations lui sont accordées si sa balise correspond à celle des ressources. Cette stratégie permet aux individus de consulter ou de modifier uniquement les AWS ressources nécessaires à leur travail.

Scénario

Supposons que vous soyez un développeur principal dans une grande entreprise nommée Exemple, ainsi qu'un administrateur IAM expérimenté. Vous savez créer et gérer des utilisateurs, des rôles et des politiques IAM. Vous voulez vous assurer que vos ingénieurs en développement et les membres de l'équipe d'assurance qualité peuvent accéder aux ressources dont ils ont besoin. Vous avez également besoin d'une stratégie qui évolue en même temps que votre entreprise.

Vous choisissez d'utiliser AWS des balises de ressources et des balises principales de rôle IAM pour mettre en œuvre une stratégie ABAC pour les services qui la prennent en charge, en commençant par. AWS Secrets Manager Pour connaître les services prenant en charge l'autorisation basée sur des balises, veuillez consulter AWS services qui fonctionnent avec IAM. Pour savoir quelles clés de condition de balisage vous pouvez utiliser dans une politique concernant les actions et les ressources de chaque service, consultez la section Actions, ressources et clés de condition pour les AWS services. Vous pouvez configurer votre fournisseur d'identité Web ou basé sur SAML pour qu'il transmette les étiquettes de session à l'interface AWS. Lorsque vos employés se fédérent dans AWS, leurs attributs sont appliqués au principal qui en résulte dans AWS. Vous pouvez ensuite utiliser l'ABAC pour autoriser ou refuser des autorisations sur la base de ces attributs. Pour savoir comment l'utilisation de balises de session avec une identité fédérée SAML diffère de ce didacticiel, veuillez consulter Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC.

Les membres de votre équipe d'ingénierie et d'assurance qualité sont sur le projet Pegasus ou le projet Unicorn. Vous sélectionnez les valeurs de balise de 3 caractères ci-dessous pour les projets et l'équipe :

- `access-project` = `peg` pour le projet Pegasus
- `access-project` = `uni` pour le projet Unicorn
- `access-team` = `eng` pour l'équipe d'ingénierie
- `access-team` = `qas` pour l'équipe d'assurance qualité

En outre, vous choisissez d'exiger l'étiquette de répartition des `cost-center` coûts pour activer les rapports AWS de facturation personnalisés. Pour plus d'informations, consultez Utilisation des balises d'allocation des coûts dans le Guide de l'utilisateur AWS Billing and Cost Management .

Résumé des décisions clés

- Les employés se connectent avec leurs informations d'identification utilisateur IAM, puis endossent le rôle IAM pour leur équipe et leur projet. Si votre entreprise possède son propre système d'identité, vous pouvez configurer une fédération pour autoriser les employés à endosser un rôle sans utilisateurs IAM. Pour plus d'informations, veuillez consulter Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC.
- La même politique est attachée à tous les rôles. Les actions sont autorisées ou refusées en fonction des balises.
- Les employés peuvent créer des ressources, mais uniquement s'ils attachent à la ressource les balises qui sont appliquées à leur rôle. Ils peuvent ainsi afficher la ressource après l'avoir créée. Les administrateurs ne sont plus tenus de mettre à jour les politiques avec l'ARN des nouvelles ressources.
- Les employés peuvent lire les ressources appartenant à leur équipe, quel que soit le projet.
- Les employés peuvent mettre à jour et supprimer des ressources appartenant à leur équipe et à leur projet.
- Les administrateurs IAM peuvent ajouter un nouveau rôle pour les nouveaux projets. Ils peuvent créer et baliser un nouvel utilisateur IAM pour lui permettre d'accéder au rôle approprié. Les administrateurs ne sont pas tenus de modifier une politique pour prendre en charge un nouveau projet ou membre de l'équipe.

Dans ce didacticiel, vous allez baliser chaque ressource, baliser les rôles de votre projet et ajouter des politiques aux rôles afin d'autoriser le comportement décrit précédemment. La politique qui en résulte autorise les rôles `Create`, `Read`, `Update` et `Delete` à accéder aux ressources qui ont les mêmes balises que le projet et l'équipe. La politique autorise également l'accès `Read` interprojet pour les ressources qui ont les mêmes balises que l'équipe.

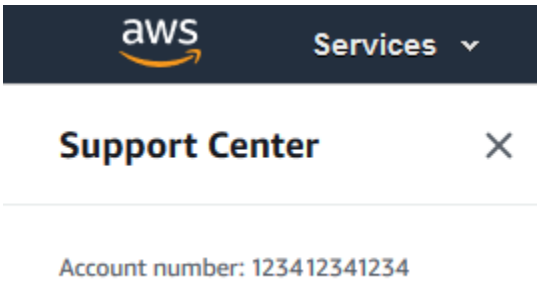
Prérequis

Pour exécuter les étapes de ce didacticiel, vous devez déjà disposer des éléments suivants :

- Et Compte AWS auquel vous pouvez vous connecter en tant qu'utilisateur disposant d'autorisations administratives.

- Votre ID de compte à 12 chiffres, que vous utilisez pour créer les rôles à l'étape 3.

  Pour trouver le numéro d'identification de votre AWS compte à l'aide du AWS Management Console, choisissez Support dans la barre de navigation en haut à droite, puis sélectionnez Support Center. Le numéro de compte (ID) apparaît dans le panneau de navigation à gauche.



- Une expérience dans la création et la modification d'utilisateurs, de rôles et de politiques IAM dans AWS Management Console. Toutefois, si vous avez besoin d'aide pour vous souvenir d'un processus de gestion IAM, ce didacticiel fournit des liens vers lesquels vous pouvez consulter step-by-step les instructions.

Étape 1 : Créer des utilisateurs test

Pour le test, créez quatre utilisateurs IAM autorisés à endosser des rôles avec les mêmes balises. Cela facilite l'ajout d'utilisateurs à vos équipes. Lorsque vous balisez les utilisateurs, ces derniers bénéficient d'un automatiquement d'un accès pour endosser le rôle. Vous n'avez pas à ajouter les utilisateurs à la politique d'approbation du rôle s'ils travaillent sur un seul projet et dans une seule équipe.

1. Créez la politique gérée par le client ci-dessous et nommez-la `access-assume-role`. Pour de plus amples informations sur la création d'une politique JSON, veuillez consulter Création de politiques IAM.

   Politique ABAC : endosser n'importe quel rôle de la politique ABAC, mais uniquement lorsque les balises de l'utilisateur et du rôle correspondent

   La politique suivante autorise un utilisateur à endosser n'importe quel rôle de votre compte avec le préfixe de nom `access-`. Le rôle doit également être balisé avec les mêmes balises de projet, d'équipe et de centre de coûts que l'utilisateur.

   Pour utiliser cette politique, remplacez le texte en italique de l'espace réservé dans l'exemple de politique par vos propres informations de compte.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TutorialAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::account-ID-without-hyphens:role/access-*",
            "Condition": {
                "StringEquals": {
                    "iam:ResourceTag/access-project": "${aws:PrincipalTag/access-
project}",
                    "iam:ResourceTag/access-team": "${aws:PrincipalTag/access-
team}",
                    "iam:ResourceTag/cost-center": "${aws:PrincipalTag/cost-
center}"
                }
            }
        }
    ]
}
```

Pour étendre ce didacticiel à un plus grand nombre d'utilisateurs, vous pouvez attacher la politique à un groupe et ajouter chaque utilisateur au groupe. Pour plus d'informations, consultez Création de groupes d'utilisateurs IAM et Ajout et suppression d'utilisateurs dans un groupe IAM.

2. Créez les utilisateurs IAM suivants, attachez la politique d'autorisation `access-assume-role`. Assurez-vous de sélectionner Fournir un accès utilisateur à la AWS Management Console, puis d'ajouter les balises suivantes. Pour de plus amples informations sur la création et le balisage d'un utilisateur, veuillez consulter Création d'utilisateurs IAM (console).

   Utilisateurs de la politique ABAC

Nom utilisateur	Clé de balise utilisateur	Valeur de balise utilisateur
`access-Arnav-peg-eng`	`access-project`	`peg`
	`access-team`	`eng`
	`cost-center`	`987654`

Nom utilisateur	Clé de balise utilisateur	Valeur de balise utilisateur
access-Mary-peg-qas	access-project access-team cost-center	peg qas 987654
access-Saanvi-uni-eng	access-project access-team cost-center	uni eng 123456
access-Carlos-uni-qas	access-project access-team cost-center	uni qas 123456

Étape 2 : Créer la politique ABAC

Créez la politique suivante et nommez-la **`access-same-project-team`**. Vous allez ajouter cette politique aux rôles ultérieurement. Pour de plus amples informations sur la création d'une politique JSON, veuillez consulter Création de politiques IAM.

Pour accéder à des politiques supplémentaires que vous pouvez adapter pour ce didacticiel, veuillez consulter les pages suivantes :

- Contrôle de l'accès pour les principaux IAM
- Amazon EC2 : autorise le démarrage ou l'arrêt des instances EC2 balisées par un utilisateur, par programmation et dans la console
- EC2 : démarrer ou arrêter les instances en fonction des balises de ressources et de principaux correspondantes
- EC2 : démarrer ou arrêter les instances en fonction des balises
- IAM : endosser des rôles qui ont une balise spécifique

Politique ABAC : accéder aux ressources Secrets Manager uniquement lorsque les balises du principal et des ressources correspondent

La politique suivante autorise les principaux à créer, lire, modifier et supprimer des ressources, mais uniquement lorsque ces ressources sont balisées avec les mêmes paires clé-valeur que le principal. Lorsqu'un principal crée une ressource, il doit ajouter les balises `access-project`, `access-team` et `cost-center` avec des valeurs qui correspondent aux balises du principal. La politique autorise également l'ajout de balises `Name` ou `OwnedBy` facultatives.

```
{
 "Version": "2012-10-17",
 "Statement": [
     {
         "Sid": "AllActionsSecretsManagerSameProjectSameTeam",
         "Effect": "Allow",
         "Action": "secretsmanager:*",
         "Resource": "*",
         "Condition": {
             "StringEquals": {
                 "aws:ResourceTag/access-project": "${aws:PrincipalTag/access-
project}",
                 "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-team}",
                 "aws:ResourceTag/cost-center": "${aws:PrincipalTag/cost-center}"
             },
             "ForAllValues:StringEquals": {
                 "aws:TagKeys": [
                     "access-project",
                     "access-team",
                     "cost-center",
                     "Name",
                     "OwnedBy"
                 ]
             },
             "StringEqualsIfExists": {
                 "aws:RequestTag/access-project": "${aws:PrincipalTag/access-project}",
                 "aws:RequestTag/access-team": "${aws:PrincipalTag/access-team}",
                 "aws:RequestTag/cost-center": "${aws:PrincipalTag/cost-center}"
             }
         }
     },
     {
         "Sid": "AllResourcesSecretsManagerNoTags",
         "Effect": "Allow",
```

```
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ReadSecretsManagerSameTeam",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:Describe*",
                "secretsmanager:Get*",
                "secretsmanager:List*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-team}"
                }
            }
        },
        {
            "Sid": "DenyUntagSecretsManagerReservedTags",
            "Effect": "Deny",
            "Action": "secretsmanager:UntagResource",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringLike": {
                    "aws:TagKeys": "access-*"
                }
            }
        },
        {
            "Sid": "DenyPermissionsManagement",
            "Effect": "Deny",
            "Action": "secretsmanager:*Policy",
            "Resource": "*"
        }
    ]
}
```

À quoi sert cette politique ?

- La instruction `AllActionsSecretsManagerSameProjectSameTeam` autorise toutes les actions de ce service sur toutes les ressources associées, mais uniquement si les balises des ressources correspondent aux balises du principal. En ajoutant `"Action": "secretsmanager:*"` à la politique, cette dernière évolue en même temps que Secrets Manager. Si Secrets Manager ajoute une nouvelle opération d'API, vous n'êtes pas obligé d'ajouter cette action à l'instruction. La instruction implémente l'ABAC en utilisant trois blocs de condition. La demande n'est autorisée que si les trois blocs renvoient la valeur « true » (vrai).
  - Le premier bloc de condition de cette instruction renvoie la valeur true si les clés de balise spécifiées sont présentes sur la ressource et que leurs valeurs correspondent aux balises du principal. Ce bloc renvoie la valeur « false » (faux) pour les étiquettes qui ne correspondent pas, ou pour les actions qui ne prennent pas en charge l'étiquettage des ressources. Pour savoir quelles actions ne sont pas autorisées par ce bloc, voir Actions, ressources et clés de condition pour AWS Secrets Manager. Cette page montre que les actions effectuées sur le type de ressource Secret prennent en charge la clé de condition `secretsmanager:ResourceTag/tag-key`. Certaines actions Secrets Manager ne prennent pas en charge ce type de ressource, y compris `GetRandomPassword` et `ListSecrets`. Vous devez créer des instructions supplémentaires pour autoriser ces actions.
  - Le deuxième bloc de condition renvoie la valeur « true » (vrai) si chaque clé d'identification transmise dans la demande figure dans la liste spécifiée. Cette opération est effectuée en utilisant `ForAllValues` avec l'opérateur de condition `StringEquals`. Si aucune clé ni sous-ensemble de l'ensemble de clés n'est transmis, la condition renvoie la valeur « true » (vrai). Cela permet les opérations `Get*` qui n'autorisent pas l'inclusion des étiquettes dans la demande. Si le demandeur inclut une clé d'identification qui ne figure pas dans la liste, la condition renvoie la valeur « false » (faux) Chaque clé de balise transmise dans la demande doit correspondre à un membre de cette liste. Pour plus d'informations, consultez Clés de contexte à valeurs multiples.
  - Le troisième bloc de condition renvoie la valeur « true » (vrai) si la demande prend en charge la transmission des étiquettes, si les trois étiquettes sont présentes et si elles correspondent aux valeurs de l'étiquette principale. Ce bloc renvoie également la valeur « true » (vrai) si la demande ne prend pas en charge la transmission des étiquettes. C'est grâce à `...IfExists` dans l'opérateur de condition. Le bloc renvoie la valeur « false » (faux) si aucune étiquette n'est transmise pendant une action qui la prend en charge, ou si les clés et les valeurs d'étiquette ne correspondent pas.
- La instruction `AllResourcesSecretsManagerNoTags` autorise les actions `GetRandomPassword` et `ListSecrets` qui ne sont pas autorisées par la première instruction.

- La instruction `ReadSecretsManagerSameTeam` autorise les opérations en lecture seule si le principal est balisé avec la même balise access-team que la ressource. Ceci est autorisé indépendamment de l'étiquette du projet ou du centre de coûts.
- La instruction `DenyUntagSecretsManagerReservedTags` refuse les demandes de suppression de balises avec des clés commençant par « access- » de Secrets Manager. Ces balises servent à contrôler l'accès aux ressources. Leur suppression pourrait supprimer des autorisations.
- La déclaration `DenyPermissionsManagement` refuse l'accès à la création, à la modification ou à la suppression de politiques basées sur les ressources de Secrets Manager. Ces politiques peuvent être utilisées pour modifier les autorisations du secret.

> **Important**
>
> Cette politique utilise une stratégie qui autorise toutes les actions d'un service, mais refuse explicitement les actions de modification des autorisations. Le refus d'une action remplace toute autre politique autorisant le principal à effectuer cette action. Ce refus peut entraîner des résultats imprévus. La bonne pratique consiste à utiliser uniquement le refus explicite lorsqu'aucune circonstance ne doit autoriser cette action. Sinon, dressez la liste des actions autorisées. Les actions indésirables seront refusées par défaut.

Étape 3 : Créer les rôles

Créez les rôles IAM suivants et attachez la politique **`access-same-project-team`** que vous avez créée à l'étape précédente. Pour plus d'informations sur la création de rôles IAM, veuillez consulter Création d'un rôle pour la délégation d'autorisations à un utilisateur IAM.. Si vous sélectionnez d'utiliser la fédération plutôt que des utilisateurs et des rôles IAM, veuillez consulter Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC.

Rôles de l'ABAC (Contrôle d'accès basé sur les attributs)

Fonction de tâche	Nom du rôle	Balises de rôle	Description du rôle
Ingénierie du projet Pegasus	`access-peg-engineering`	`access-project = peg`	Autorise les ingénieurs à lire toutes les ressources d'ingénierie et à créer et gérer les

Fonction de tâche	Nom du rôle	Balises de rôle	Description du rôle
		`access-team = eng` `cost-center = 987654`	ressources d'ingénierie du projet Pegasus.

Assurance qualité du projet Pegasus	`access-peg-quality-assurance`	`access-project = peg`
		`access-team = qas`
		`cost-center = 987654`
			Autorise l'équipe d'assurance qualité à lire toutes les ressources d'assurance qualité et à créer et gérer toutes les ressources d'assurance qualité du projet Pegasus.

Ingénierie du projet Unicorn	`access-uni-engineering`	`access-project = uni`
		`access-team = eng`
		`cost-center = 123456`
			Autorise les ingénieurs à lire toutes les ressources d'ingénierie et à créer et gérer les ressources d'ingénierie du projet Unicorn.

Fonction de tâche	Nom du rôle	Balises de rôle	Description du rôle

Assurance qualité du projet Unicorn	`access-uni-quality-assurance`	`access-project = uni`
		
		`access-team = qas`
		
		`cost-center = 123456`
			Autorise l'équipe d'assurance qualité à lire toutes les ressources d'assurance qualité et à créer et gérer toutes les ressources d'assurance qualité du projet Unicorn.

Étape 4 : Tester la création de secrets

La politique d'autorisation attachée aux rôles autorise les employés à créer des secrets. Ceci n'est autorisé que si le secret est labelisé au niveau du projet, de l'équipe et du centre de coûts. Vérifiez que vos autorisations fonctionnent comme prévu en vous connectant comme vos utilisateurs, en endossant le bon rôle et en testant l'activité dans Secrets Manager.

Pour tester la création d'un secret avec et sans les balises requises

1. Dans la fenêtre principale de votre navigateur, restez connecté en tant qu'utilisateur administrateur afin que vous puissiez passer en revue les utilisateurs, les rôles et les politiques dans IAM. Utilisez une fenêtre de navigation privée ou un navigateur séparé pour votre test. Connectez-vous ensuite en tant que l'utilisateur IAM `access-Arnav-peg-eng`, puis ouvrez la console Secrets Manager à l'adresse https://console.aws.amazon.com/secretsmanager/.
2. Tentez de basculer vers le rôle `access-uni-engineering`.

   Cette opération échoue, car les valeurs de balise `access-project` et `cost-center` ne correspondent pas à l'utilisateur `access-Arnav-peg-eng` et au rôle `access-uni-engineering`.

   Pour plus d'informations sur le changement de rôle dans le AWS Management Console, voir Changement de rôle (console)
3. Basculez vers le rôle `access-peg-engineering`.

4. Enregistrez un nouveau secret en utilisant les informations suivantes. Pour savoir comment enregistrer un secret, veuillez consulter Création d'un secret basique dans le Guide de l'utilisateur AWS Secrets Manager .

> **Important**
>
> Secrets Manager affiche des alertes indiquant que vous ne disposez pas d'autorisations pour des services AWS supplémentaires qui fonctionnent avec Secrets Manager. Par exemple, pour créer des informations d'identification pour une base de données Amazon RDS, vous devez être autorisé à décrire des instances RDS, des clusters RDS et des clusters Amazon Redshift. Vous pouvez ignorer ces alertes car vous n'utilisez pas ces AWS services spécifiques dans ce didacticiel.

1. Dans la section Sélectionner un type de secret sélectionnez Autre type de secrets. Dans les deux zones de texte, saisissez `test-access-key` et `test-access-secret`.
2. Saisissez `test-access-peg-eng` dans le champ Nom du secret .
3. Ajoutez différentes combinaisons de balises du tableau suivant et visualisez le comportement attendu.
4. Choisissez Stocker pour tenter de créer le secret. Si le stockage échoue, revenez aux pages précédentes de la console Secrets Manager et utilisez l'ensemble de balises suivant dans le tableau ci-dessous. Le dernier ensemble de balises est autorisé et crée le secret avec succès.

Combinaisons de balises ABAC pour le rôle **`test-access-peg-eng`**

`access-project` Valeur de l'étiquette	**`access-team`** Valeur de l'étiquette	**`cost-center`** Valeur de l'étiquette	Balises supplémentaires	Comportement attendu
(aucun)	(aucun)	(aucun)	(aucun)	Refusé, car la valeur de la balise `access-project` ne correspond pas à la valeur `peg` du rôle.

access-project Valeur de l'étiquette	**access-team** Valeur de l'étiquette	**cost-center** Valeur de l'étiquette	Balises supplémentaires	Comportement attendu
uni	eng	987654	(aucun)	Refusé, car la valeur de la balise access-project ne correspond pas à la valeur peg du rôle.
peg	qas	987654	(aucun)	Refusé, car la valeur de la balise access-team ne correspond pas à la valeur eng du rôle.
peg	eng	123456	(aucun)	Refusé, car la valeur de la balise cost-center ne correspond pas à la valeur 987654 du rôle.
peg	eng	987654	owner = Jane	Refusé, car la balise owner supplémentaire n'est pas autorisée par la politique, même si les trois balises requises sont présentes et que leurs valeurs correspondent à celles du rôle.
peg	eng	987654	Name = Jane	Autorisé, car les trois balises obligatoires sont présentes et leurs valeurs correspondent aux valeurs du rôle. Vous êtes également autorisé à inclure la balise Name facultative.

5. Déconnectez-vous et répétez les trois premières étapes de cette procédure pour chaque rôle et chaque valeur de balise ci-dessous. À la quatrième étape de cette procédure, testez tous les ensembles de balises manquantes, de balises facultatives, de balises non autorisées et de valeurs de balises non valides de votre choix. Puis, utilisez les balises requises pour créer un secret avec les balises et le nom suivants.

Rôles et balises ABAC

Nom utilisateur	Nom du rôle	Nom du secret	Balises du secret
access-Mary-peg-qas	access-peg-quality-assurance	test-access-peg-qas	access-project = peg access-team = qas cost-center = 987654
access-Saanvi-uni-eng	access-uni-engineering	test-access-uni-eng	access-project = uni access-team = eng cost-center = 123456
access-Carlos-uni-qas	access-uni-quality-assurance	test-access-uni-qas	access-project = uni access-team = qas cost-center = 123456

Étape 5 : Tester l'affichage des secrets

La politique que vous avez attachée à chaque rôle autorise les employés à afficher tous les secrets labelisés avec leur nom d'équipe, quel que soit leur projet. Vérifiez que vos autorisations fonctionnent comme prévu en testant vos rôles dans Secrets Manager.

Pour tester l'affichage d'un secret avec et sans les balises requises

1. Connectez-vous en tant que l'un des utilisateurs IAM suivants :
   - `access-Arnav-peg-eng`
   - `access-Mary-peg-qas`
   - `access-Saanvi-uni-eng`
   - `access-Carlos-uni-qas`
2. Basculez vers le rôle correspondant :
   - `access-peg-engineering`
   - `access-peg-quality-assurance`
   - `access-uni-engineering`
   - `access-uni-quality-assurance`

   Pour plus d'informations sur le changement de rôle dans le AWS Management Console, consultezChangement de rôle (console).
3. Dans le panneau de navigation de gauche, sélectionnez l'icône de menu permettant de développer le menu, puis sélectionnez Secrets.
4. Vous devriez voir les quatre secrets du tableau, quel que soit votre rôle actuel. C'est le comportement attendu, car la politique nommée `access-same-project-team` autorise l'action `secretsmanager:ListSecrets` pour toutes les ressources.
5. Choisissez l'un des secrets.
6. Sur la page détaillée du secret, les étiquettes de votre rôle déterminent si vous pouvez afficher le contenu de la page. Comparez le nom de votre rôle avec celui de votre secret. S'ils ont le même nom d'équipe, alors les balises `access-team` correspondent. Si elles ne correspondent pas, l'accès est refusé.

   Comportement d'affichage du secret ABAC pour chaque rôle

Nom du rôle	Nom du secret	Comportement attendu
`access-peg-engineering`	`test-access-peg-eng`	Autorisé
	`test-access-peg-qas`	Refusé

Nom du rôle	Nom du secret	Comportement attendu
	`test-access-uni-eng`	Autorisé
	`test-access-uni-qas`	Refusé
`access-peg-quality-assurance`	`test-access-peg-eng`	Refusé
	`test-access-peg-qas`	Autorisé
	`test-access-uni-eng`	Refusé
	`test-access-uni-qas`	Autorisé
`access-uni-engineering`	`test-access-peg-eng`	Autorisé
	`test-access-peg-qas`	Refusé
	`test-access-uni-eng`	Autorisé
	`test-access-uni-qas`	Refusé
`access-uni-quality-assurance`	`test-access-peg-eng`	Refusé
	`test-access-peg-qas`	Autorisé
	`test-access-uni-eng`	Refusé
	`test-access-uni-qas`	Autorisé

7. Dans les miniatures en haut de la page, sélectionnez Secrets pour revenir à la liste des secrets. Répétez les étapes de cette procédure en utilisant différents rôles pour vérifier si vous pouvez afficher chacun des secrets.

Étape 6 : Tester l'adaptabilité

L'évolutivité est une raison majeure de privilégier le contrôle d'accès basé sur les attributs (ABAC) par rapport au contrôle d'accès basé sur les rôles (RBAC). Au fur et à mesure que votre entreprise ajoute de nouveaux projets, équipes ou personnes AWS, vous n'avez pas besoin de mettre à jour vos politiques basées sur l'ABAC. Par exemple, supposons que la société Exemple finance un nouveau projet, dont le nom de code est Centaur. Un ingénieur nommé Saanvi Sarkar sera l'ingénieur principal

du projet Centaur tout en continuant à travailler sur le projet Unicorn . Saanvi passera également en revue les travaux du projet Pegasus. Plusieurs ingénieurs embauchés récemment, dont Nikhil Jayashankar, travailleront exclusivement sur le projet Centaur .

Pour ajouter le nouveau projet à AWS

1. Connectez-vous en tant qu'utilisateur administrateur IAM, puis ouvrez la console IAM à partir de l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, sélectionnez Roles (Rôles) et ajoutez un rôle IAM nommé `access-cen-engineering`. Attachez la politique d'autorisation **`access-same-project-team`** au rôle et ajoutez les balises de rôle suivantes :
   - `access-project` = `cen`
   - `access-team` = `eng`
   - `cost-center` = `101010`
3. Dans le panneau de navigation de gauche, sélectionnez Users (Utilisateurs).
4. Ajoutez un nouvel utilisateur nommé `access-Nikhil-cen-eng`, attachez la politique nommée `access-assume-role`, et ajoutez les balises utilisateur suivantes.
   - `access-project` = `cen`
   - `access-team` = `eng`
   - `cost-center` = `101010`
5. Utilisez les procédures des sections Étape 4 : Tester la création de secrets et Étape 5 : Tester l'affichage des secrets. Dans une autre fenêtre de navigateur, vérifiez que Nikhil ne peut créer que des secrets d'ingénierie pour le projet Centaur et qu'il peut afficher tous les secrets d'ingénierie.
6. Dans la fenêtre principale du navigateur à partir de laquelle vous vous êtes connecté en tant qu'administrateur, sélectionnez l'utilisateur `access-Saanvi-uni-eng`.
7. Dans l'onglet Autorisations, supprimez la politique access-assume-roled'autorisations.
8. Ajoutez la politique en ligne ci-dessous et nommez-la `access-assume-specific-roles`. Pour de plus amples informations sur l'intégration d'une politique en ligne pour un utilisateur, veuillez consulter Pour intégrer une politique en ligne pour un utilisateur ou un rôle (console).

   Politique ABAC : endosser uniquement des rôles spécifiques

Cette politique autorise Saanvi à assumer les rôles d'ingénierie pour les projets Pegasus et Centaure. Il est nécessaire de créer cette politique personnalisée, car IAM ne prend pas en charge les balises à valeurs multiples. Vous ne pouvez pas baliser l'utilisateur Saanvi avec `access-project` = `peg` et `access-project` = `cen`. De plus, le modèle AWS d'autorisation ne peut pas correspondre aux deux valeurs. Pour plus d'informations, consultez Règles de balisage dans IAM et AWS STS. Vous devez spécifier manuellement les deux rôles qu'elle peut endosser.

Pour utiliser cette politique, remplacez le texte en italique de l'espace réservé dans l'exemple de politique par vos propres informations de compte.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TutorialAssumeSpecificRoles",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": [
                "arn:aws:iam::account-ID-without-hyphens:role/access-peg-
engineering",
                "arn:aws:iam::account-ID-without-hyphens:role/access-cen-
engineering"
            ]
        }
    ]
}
```

9. Utilisez les procédures des sections Étape 4 : Tester la création de secrets et Étape 5 : Tester l'affichage des secrets. Dans une autre fenêtre du navigateur, vérifiez que Saanvi peut endosser les deux rôles. Vérifiez qu'elle peut créer des secrets uniquement pour son projet, son équipe et son centre de coûts, en fonction des balises du rôle. Vérifiez également qu'elle peut afficher les détails de tous les secrets appartenant à l'équipe d'ingénierie, y compris ceux qu'elle vient de créer.

Étape 7 : Tester la mise à jour et la suppression des secrets

La politique `access-same-project-team` attachée aux rôles autorise les employés à mettre à jour et à supprimer tous les secrets possédant les mêmes balises que leur projet, leur équipe et leur

centre de coûts. Vérifiez que vos autorisations fonctionnent comme prévu en testant vos rôles dans Secrets Manager.

Pour tester la mise à jour et la suppression d'un secret avec et sans les balises requises

1. Connectez-vous en tant que l'un des utilisateurs IAM suivants :
   - `access-Arnav-peg-eng`
   - `access-Mary-peg-qas`
   - `access-Saanvi-uni-eng`
   - `access-Carlos-uni-qas`
   - `access-Nikhil-cen-eng`
2. Basculez vers le rôle correspondant :
   - `access-peg-engineering`
   - `access-peg-quality-assurance`
   - `access-uni-engineering`
   - `access-peg-quality-assurance`
   - `access-cen-engineering`

   Pour plus d'informations sur le changement de rôle dans le AWS Management Console, consultezChangement de rôle (console).
3. Pour chaque rôle, mettez à jour la description du secret, puis supprimez les secrets suivants. Pour de plus amples informations, veuillez consulter Modification d'un secret et Suppression et restauration d'un secret dans le Guide de l'utilisateur AWS Secrets Manager .

   Comportement de mise à jour et de suppression du secret ABAC pour chaque rôle

Nom du rôle	Nom du secret	Comportement attendu
`access-peg-engineering`	`test-access-peg-eng`	Autorisé
	`test-access-uni-eng`	Refusé
	`test-access-uni-qas`	Refusé

Nom du rôle	Nom du secret	Comportement attendu
`access-peg-quality-assurance`	`test-access-peg-qas`	Autorisé
	`test-access-uni-eng`	Refusé
`access-uni-engineering`	`test-access-uni-eng`	Autorisé
	`test-access-uni-qas`	Refusé
`access-peg-quality-assurance`	`test-access-uni-qas`	Autorisé

Récapitulatif

Vous avez à présent terminé toutes les étapes nécessaires pour utiliser des balises pour le contrôle d'accès basé sur les attributs (ABAC). Vous avez appris à définir une stratégie de balisage. Vous avez appliqué cette stratégie à vos principaux et à vos ressources. Vous avez créé et appliqué une politique qui met exécute la stratégie pour Secrets Manager. Vous avez également appris que l'ABAC évolue facilement lorsque vous ajoutez de nouveaux projets et membres d'équipe. Désormais, vous pouvez vous connecter à la console IAM avec vos rôles test et tester l'utilisation de balises ABAC dans AWS.

> **Note**
>
> Vous avez ajouté des politiques qui autorisent les actions uniquement dans des conditions spécifiques. Si vous appliquez une politique différente à vos utilisateurs ou rôles disposant d'autorisations plus larges, il se peut que les actions ne soient pas limitées pour demander le balisage. Par exemple, si vous accordez à un utilisateur des autorisations administratives complètes à l'aide de la politique `AdministratorAccess` AWS gérée, ces politiques ne limitent pas cet accès. Pour plus d'informations sur la façon dont les autorisations sont déterminées lorsque plusieurs politiques sont impliquées, veuillez consulter Identification d'une demande autorisée ou refusée dans un compte.

Ressources connexes

Pour plus d'informations, consultez les ressources suivantes :

- À quoi sert ABAC ? AWS
- AWS clés contextuelles de condition globale
- Création d'utilisateurs IAM (console)
- Création d'un rôle pour la délégation d'autorisations à un utilisateur IAM.
- Balisage des ressources IAM
- Contrôle de l'accès aux AWS ressources à l'aide de balises
- Changement de rôle (console)
- Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC

Pour savoir comment surveiller les balises de votre compte, consultez Surveiller les modifications des balises sur les AWS ressources avec des flux de travail sans serveur et Amazon CloudWatch Events.

Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC

Le contrôle d’accès par attributs (ABAC) est une stratégie d’autorisation qui définit des autorisations en fonction des attributs. Dans AWS, ces attributs sont appelés balises. Vous pouvez associer des balises aux ressources IAM, notamment aux entités IAM (utilisateurs ou rôles), et aux AWS ressources. Lorsque les entités sont utilisées pour envoyer des demandes AWS, elles deviennent des entités principales et ces entités incluent des balises.

Vous pouvez également transmettre des balises de session lorsque vous endossez un rôle ou fédérez un utilisateur. Vous pouvez ensuite définir des politiques qui utilisent des clés de condition de balise pour accorder des autorisations à vos principaux en fonction de leurs balises. Lorsque vous utilisez des balises pour contrôler l'accès à vos ressources AWS , vous autorisez vos équipes et vos ressources à se développer en modifiant moins les politiques AWS . Les politiques ABAC sont plus flexibles que AWS les politiques traditionnelles, qui vous obligent à répertorier chaque ressource individuelle. Pour de plus amples informations sur l'ABAC et ses avantages par rapport aux politiques traditionnelles, veuillez consulter À quoi sert ABAC ? AWS.

Si votre entreprise utilise un fournisseur d'identité (IdP, identity provider) basé sur SAML pour gérer les identités des utilisateurs de l'entreprise, vous pouvez utiliser les attributs SAML pour un contrôle d'accès affiné dans l’interface AWS. Les attributs peuvent inclure des identifiants de centre de coûts, des adresses e-mail d'utilisateurs, des classifications de services et des affectations de projet. Lorsque vous transmettez ces attributs en tant qu’étiquettes de session, vous pouvez ensuite contrôler l'accès à l’interface AWS en fonction de ces étiquettes de session.

Pour réaliser le didacticiel de l'ABAC en transmettant les attributs SAML à votre principal de session, effectuez les tâches de la section Tutoriel IAM : définir les autorisations d'accès aux AWS ressources en fonction des balises avec les modifications citées dans cette rubrique.

Prérequis

Pour réaliser les étapes d'utilisation des étiquettes de session SAML pour l'ABAC, vous devez déjà disposer des éléments suivants :

- Un accès à un fournisseur d'identité basé sur SAML vous permettant de créer des utilisateurs test avec des attributs spécifiques.
- La possibilité de se connecter en tant qu'utilisateur disposant d'autorisations d'administrateur.
- Une expérience dans la création et la modification d'utilisateurs, de rôles et de politiques IAM dans AWS Management Console. Toutefois, si vous avez besoin d'aide pour vous souvenir d'un processus de gestion IAM, le didacticiel ABAC fournit des liens vers lesquels vous pouvez consulter step-by-step les instructions.
- Une expérience dans la configuration d'un fournisseur d'identité SAML dans IAM. Pour afficher de plus amples informations et des liens vers la documentation détaillée IAM, veuillez consulter Transmission de balises de session à l'aide de AssumeRoleWith SAML.

Étape 1 : Créer des utilisateurs test

Ignorez les instructions de la section Étape 1 : Créer des utilisateurs test. Étant donné que vos identités sont définies dans votre fournisseur, il n'est pas nécessaire que vous ajoutiez des utilisateurs IAM pour vos employés.

Étape 2 : Créer la politique ABAC

Suivez les instructions de la section Étape 2 : Créer la politique ABAC pour créer la politique gérée spécifiée dans IAM.

Étape 3 : Créer et configurer le rôle SAML

Lorsque vous utilisez le didacticiel ABAC pour SAML, vous devez effectuer des étapes supplémentaires pour créer le rôle, configurer l'IdP SAML et activer l'accès. AWS Management Console Pour plus d’informations, consultez Étape 3 : Créer les rôles.

Étape 3A : Créer le rôle SAML

Créez un rôle unique qui approuve votre fournisseur d'identité SAML et l'utilisateur `test-session-tags` que vous avez créé à l'étape 1. Le didacticiel de l'ABAC utilise des rôles distincts avec des balises de rôle différentes. Étant donné que vous transmettez des balises de session à partir de votre fournisseur d'identité SAML, vous n'avez besoin que d'un seul rôle. Pour savoir comment créer un rôle basé sur SAML, veuillez consulter Création d'un rôle pour la fédération SAML 2.0 (console).

Nommez le rôle `access-session-tags`. Attachez la politique d'autorisation `access-same-project-team` au rôle. Modifiez la politique d'approbation de rôle pour utiliser la politique suivante. Pour obtenir des instructions détaillées sur la modification de la relation d'approbation d'un rôle, veuillez consulter Modification d'un rôle (console).

La politique d'approbation de rôle suivante permet à votre fournisseur d'identité SAML et à l'utilisateur `test-session-tags` d'endosser le rôle. Lorsqu'ils endossent le rôle, ils doivent transmettre les trois balises de session spécifiées. L'action `sts:TagSession` est requise pour autoriser la transmission des balises de session.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSamlIdentityAssumeRole",
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRoleWithSAML",
                "sts:TagSession"
            ],
            "Principal": {"Federated":"arn:aws:iam::123456789012:saml-
provider/ExampleCorpProvider"},
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/cost-center": "*",
                    "aws:RequestTag/access-project": "*",
                    "aws:RequestTag/access-team": [
                        "eng",
                        "qas"
                    ]
                },
                "StringEquals": {"SAML:aud": "https://signin.aws.amazon.com/saml"}
            }
        }
```

```
    ]
}
```

La `AllowSamlIdentityAssumeRole` déclaration permet aux membres des équipes d'ingénierie et d'assurance qualité d'assumer ce rôle lorsqu'ils se fédérent avec l'IdP AWS d'Example Corporation. Le fournisseur SAML `ExampleCorpProvider` est défini dans IAM. L'administrateur a déjà configuré l'assertion SAML pour transmettre les trois balises de session requises. L'assertion peut transmettre des balises supplémentaires, mais ces trois balises doivent être présentes. Les attributs de l'identité peuvent avoir n'importe quelle valeur pour les balises `cost-center` et `access-project`. Toutefois, la valeur de l'attribut `access-team` doit correspondre à `eng` ou `qas` pour indiquer que l'identité se trouve dans l'équipe d'ingénierie ou d'assurance qualité.

Étape 3B : Configurer le fournisseur d'identité SAML

Configurez votre fournisseur d'identité SAML pour qu'il transmette les attributs `cost-center`, `access-project` et `access-team` en tant que balises de session. Pour de plus amples informations, veuillez consulter Transmission de balises de session à l'aide de AssumeRoleWith SAML.

Pour transmettre ces attributs en tant que balises de session, incluez les éléments suivants dans votre assertion SAML.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:cost-center">
  <AttributeValue>987654</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:access-project">
  <AttributeValue>peg</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:access-team">
  <AttributeValue>eng</AttributeValue>
</Attribute>
```

Étape 3C : Activer l'accès à la console

Activez l'accès à la console pour vos utilisateurs SAML fédérés. Pour de plus amples informations, veuillez consulter Permettre aux utilisateurs fédérés SAML 2.0 d'accéder au AWS Management Console.

Étape 4 : Tester la création de secrets

Fédérez-vous pour AWS Management Console utiliser le `access-session-tags` rôle. Pour plus d'informations, consultez Permettre aux utilisateurs fédérés SAML 2.0 d'accéder au AWS Management Console. Puis, suivez les instructions de la section Étape 4 : Tester la création de secrets pour créer des secrets. Utilisez différentes identités SAML avec des attributs pour correspondre aux balises indiquées dans le didacticiel de l'ABAC. Pour plus d'informations, veuillez consulter Étape 4 : Tester la création de secrets.

Étape 5 : Tester l'affichage des secrets

Suivez les instructions de la section Étape 5 : Tester l'affichage des secrets pour afficher les secrets que vous avez créés à l'étape précédente. Utilisez différentes identités SAML avec des attributs pour correspondre aux balises indiquées dans le didacticiel de l'ABAC.

Étape 6 : Tester l'adaptabilité

Suivez les instructions de la section Étape 6 : Tester l'adaptabilité pour tester l'évolutivité. Pour ce faire, ajoutez une nouvelle identité à votre fournisseur d'identité basé sur SAML avec les attributs suivants :

- `cost-center = 101010`
- `access-project = cen`
- `access-team = eng`

Étape 7 : Tester la mise à jour et la suppression des secrets

Suivez les instructions de l'étape Étape 7 : Tester la mise à jour et la suppression des secrets pour mettre à jour et supprimer des secrets. Utilisez différentes identités SAML avec des attributs pour correspondre aux balises indiquées dans le didacticiel de l'ABAC.

> **Important**
>
> Supprimez tous les secrets que vous avez créés pour éviter les frais de facturation. Pour de plus amples informations sur la tarification de Secrets Manager, veuillez consulter Tarification AWS Secrets Manager.

Récapitulatif

Vous avez maintenant terminé avec succès toutes les étapes nécessaires pour utiliser les balises de session SAML et les balises de ressources pour la gestion des autorisations.

> **Note**
>
> Vous avez ajouté des politiques qui autorisent les actions uniquement dans des conditions spécifiques. Si vous appliquez une politique différente à vos utilisateurs ou rôles disposant d'autorisations plus larges, il se peut que les actions ne soient pas limitées pour demander le balisage. Par exemple, si vous accordez à un utilisateur des autorisations administratives complètes à l'aide de la politique `AdministratorAccess` AWS gérée, ces politiques ne limitent pas cet accès. Pour plus d'informations sur la façon dont les autorisations sont déterminées lorsque plusieurs politiques sont impliquées, veuillez consulter Identification d'une demande autorisée ou refusée dans un compte.

Didacticiel IAM : permettre aux utilisateurs de gérer leurs informations d'identification et leurs paramètres MFA

Vous pouvez autoriser vos utilisateurs à gérer leurs propres dispositifs d'authentification multifactorielle (MFA) et leurs informations d'identification sur la page Informations d'identification de sécurité. Vous pouvez utiliser l' AWS Management Console pour configurer des informations d'identification (clés d'accès, mots de passe, certificats de signature et clés publiques SSH), supprimer ou désactiver les informations d'identification qui ne sont pas nécessaires et activer des périphériques MFA pour vos utilisateurs. Cette méthode est utile pour un petit nombre d'utilisateurs, mais la tâche peut rapidement devenir fastidieuse si le nombre d'utilisateurs augmente. Ce didacticiel vous montre comment mettre en place ces bonnes pratiques sans surcharger de travail vos administrateurs.

Ce didacticiel explique comment autoriser les utilisateurs à accéder aux AWS services, mais uniquement lorsqu'ils se connectent à l'aide de la MFA. S'ils ne sont pas connectés avec un dispositif MFA, les utilisateurs ne peuvent pas accéder à d'autres services.

Ce flux de travail se compose de trois étapes de base.

[Étape 1 : Créer une politique pour appliquer l'authentification MFA]

Créez une politique gérée par le client qui interdit toutes les actions à l'exception des quelques actions IAM. Ces exceptions permettent à un utilisateur de modifier ses propres informations d'identification et de gérer ses appareils MFA sur la page Informations d'identification de sécurité. Plus d'informations sur la manière d'accéder à cette page, veuillez consulter Comment les utilisateurs IAM modifient leur mot de passe (console).

[Étape 2 : Attacher des politiques à votre groupe d'utilisateurs test]

Créez un groupe d'utilisateurs dont les membres ont un accès total à toutes les actions Amazon EC2 s'ils se connectent avec MFA. Pour créer un tel groupe d'utilisateurs, vous devez associer à la fois la politique AWS gérée appelée `AmazonEC2FullAccess` et la politique gérée par le client que vous avez créée lors de la première étape.

[Étape 3 : Test de l'accès utilisateur]

Connectez-vous en tant qu'utilisateur test pour vérifier que l'accès à Amazon EC2 est bloqué jusqu'à ce que l'utilisateur crée un dispositif MFA. L'utilisateur peut alors se connecter à l'aide de ce dispositif.

Prérequis

Pour exécuter les étapes de ce didacticiel, vous devez déjà disposer des éléments suivants :

- Et Compte AWS auquel vous pouvez vous connecter en tant qu'utilisateur IAM avec des autorisations administratives.
- Votre ID de compte, que vous entrez dans la politique à l'étape 1.

  Pour trouver votre ID de compte, sur la barre de navigation située en haut de la page, sélectionnez Support, puis sélectionnez Support Center (Centre de support). L'ID de compte se trouve dans le menu Support de cette page.
- Un périphérique MFA virtuel (logiciel), une clé de sécurité FIDO ou un dispositif MFA matériel.
- Un utilisateur IAM test et membre du groupe d'utilisateurs suivant :

Créer un utilisateur		Créer et configurer le compte d'un groupe d'utilisateurs		
Nom utilisateur	Autres instructions	Nom du groupe d'utilisateurs	Ajouter l'utilisateur en tant que membre	Autres instructions
MFAUser	Choisissez uniquement l'option pour Activer l'accès à la console (facultatif) et attribuez un mot de passe.	EC2MFA	MFAUser	N'attachez PAS de politique ni n'accordez d'autorisations à ce groupe d'utilisateurs.

Étape 1 : Créer une politique pour appliquer l'authentification MFA

Vous commencez par créer une politique gérée par le client IAM qui refuse toutes les autorisations, sauf celles requises par les utilisateurs IAM pour gérer leurs informations d'identification et appareils MFA.

1. Connectez-vous à la console de AWS gestion en tant qu'utilisateur avec des informations d'identification d'administrateur. Pour respecter les meilleures pratiques IAM, ne vous connectez pas avec vos Utilisateur racine d'un compte AWS informations d'identification.

   > **Important**
   >
   > Les meilleures pratiques IAM recommandent de demander aux utilisateurs humains d'utiliser la fédération avec un fournisseur d'identité pour accéder à l'aide d'informations d'identification temporaires plutôt que d' AWS utiliser des utilisateurs IAM dotés d'informations d'identification à long terme.

2. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
3. Dans le panneau de navigation, sélectionnez Policies (Politiques), puis Create policy (Créer une politique).
4. Sélectionnez l'onglet JSON, et copiez le texte du document de politique JSON suivant : AWS: permet aux utilisateurs IAM authentifiés par MFA de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

5. Collez cette politique dans la zone de texte JSON. Résolvez tous les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant.

   > **Note**
   >
   > Vous pouvez basculer à tout moment entre les options Éditeur visuel et JSON. Cependant, la politique ci-dessus inclut l'élément `NotAction` qui n'est pas pris en charge dans l'éditeur visuel. Pour cette politique, vous verrez une notification dans l'onglet Visual editor (Éditeur visuel). Revenez à JSON pour continuer à travailler avec cette politique.
   >
   > Cet exemple de politique n'autorise pas les utilisateurs à réinitialiser un mot de passe lorsqu'ils se connectent à l' AWS Management Console pour la première fois. Nous vous recommandons de n'accorder des autorisations aux nouveaux utilisateurs qu'après leur enregistrement et aient réinitialisé leur mot de passe.

6. Sur la page Vérifier et créer, tapez **`Force_MFA`** pour le nom de la politique. Pour la description de la politique, tapez **`This policy allows users to manage their own passwords and MFA devices but nothing else unless they authenticate with MFA.`** dans la zone Balises ; vous pouvez éventuellement ajouter des paires clé-valeur de balises à la politique gérée par le client. Vérifiez les autorisations accordées par votre politique, puis choisissez Créer une politique pour enregistrer votre travail.

   La nouvelle politique s'affiche dans la liste des politiques gérées et est prête à être attachée.

Étape 2 : Attacher des politiques à votre groupe d'utilisateurs test

Vous allez maintenant attacher deux politiques à votre groupe d'utilisateurs IAM test. Elles seront utilisées pour octroyer des autorisations protégées par MFA.

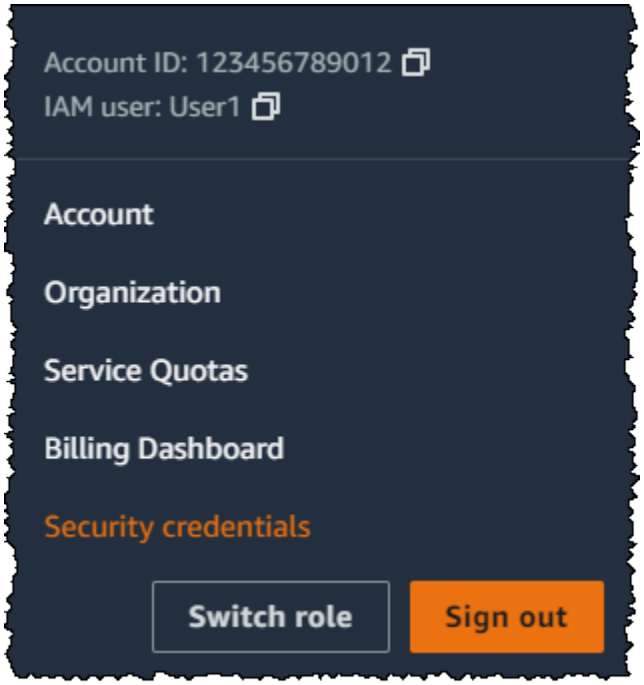
1. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs).
2. Dans la zone de recherche, saisissez **`EC2MFA`**, puis sélectionnez le nom du groupe (et pas la case à cocher) dans la liste.
3. Choisissez l'onglet Autorisations, puis Ajouter des autorisations, et enfin Attacher des politiques.
4. Sur la page Attach permission policies to EC2MFA group (Joindre des politiques d'autorisation au groupe EC2MFA), dans la zone de recherche, saisissez **`EC2Full`**. Cochez ensuite la case à côté d'AmazonEC2 FullAccess dans la liste. N'enregistrez pas vos modifications pour le moment.

5. Dans la zone de recherche, saisissez **`Force`**, puis cochez la case située en regard de Force_MFA dans la liste.
6. Sélectionnez Attach Policies (Attacher des politiques).

Étape 3 : Test de l'accès utilisateur

Dans cette étape du didacticiel, vous vous connectez en tant qu'utilisateur test afin de vérifier que la politique fonctionne comme prévu.

1. Connectez-vous à votre annonce **`MFAUser`** avec Compte AWS le mot de passe que vous avez attribué dans la section précédente. Utilisez l'URL : `https://<alias or account ID number>.signin.aws.amazon.com/console`
2. Sélectionnez EC2 pour ouvrir la console Amazon EC2 et vérifiez que l'utilisateur ne dispose d'aucune autorisation.
3. Dans la barre de navigation en haut à droite, sélectionnez votre nom d'utilisateur `MFAUser`, puis Security Credentials (Informations d'identification de sécurité).



4. Maintenant, ajoutez un dispositif MFA. Dans la section Multi-Factor Authentication (MFA) (Authentification multi-facteurs (MFA)) sélectionnez Assign MFA device (Attribuer un dispositif MFA).

> **Note**
>
> Vous pouvez recevoir une erreur que vous n'êtes pas autorisé à effectuer `iam:DeleteVirtualMFADevice`. Cela peut se produire si quelqu'un a commencé précédemment à attribuer un dispositif MFA virtuel pour cet utilisateur et a annulé le processus. Pour continuer, vous ou un autre administrateur devez supprimer le dispositif MFA virtuel non attribué existant de l'utilisateur. Pour plus d'informations, consultez Je ne suis pas autorisé à effectuer : iam : MFAdevice DeleteVirtual.

5. Dans le cadre de ce didacticiel, nous utilisons un appareil MFA (basé sur un logiciel) virtuel, comme l'application Google Authenticator sur un téléphone portable. Choisissez l'application Authenticator, puis cliquez sur Next (Suivant).

   IAM génère et affiche les informations de configuration du dispositif MFA virtuel, notamment un graphique de code QR. Le graphique est une représentation de la clé de configuration secrète que l'on peut saisir manuellement sur des dispositifs qui ne prennent pas en charge les codes QR.

6. Ouvrez votre application MFA virtuelle. (Pour obtenir une liste des applications que vous pouvez utiliser pour héberger des dispositifs MFA virtuels, veuillez consulter Applications MFA virtuelles.) Si l'application MFA virtuelle prend en charge plusieurs comptes (plusieurs dispositifs MFA virtuels), sélectionnez l'option permettant de créer un compte (un nouveau dispositif MFA virtuel).

7. Déterminez si l'application MFA prend en charge les codes QR, puis effectuez l'une des actions suivantes :

   - Dans l'assistant, sélectionnez Show QR code (Afficher le code QR). Utilisez ensuite l'application pour analyser le code QR. Par exemple, vous pouvez choisir l'icône de caméra ou une option similaire à Scan code, puis utiliser la caméra du dispositif pour analyser le code.
   - Dans l'assistant Set up device (Configurer le dispositif), sélectionnez Show secret key (Afficher la clé secrète), puis saisissez la clé secrète dans votre application MFA.

   Une fois que vous avez terminé, le dispositif MFA virtuel commence à générer des mots de passe uniques.

8. Dans l'assistant Configurer le dispositif, dans la zone Saisir le code à partir de votre application d'authentification, saisissez le mot de passe unique qui s'affiche actuellement sur le dispositif MFA virtuel. Sélectionnez Register MFA (Enregistrer le dispositif MFA).

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré le code. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la requête, l'appareil MFA est associé avec succès à l'utilisateur. Cependant, l'appareil MFA n'est pas synchronisé. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période. Dans ce cas, vous pouvez resynchroniser le dispositif.

Le périphérique MFA virtuel est maintenant prêt à être utilisé avec. AWS

9. Déconnectez-vous de la console, puis reconnectez-vous en tant que **`MFAUser`**. Cette fois, vous AWS êtes invité à saisir un code MFA sur votre téléphone. Lorsque vous recevez le code, entrez-le dans le champ et sélectionnez Submit (Soumettre).
10. Sélectionnez EC2 pour rouvrir la console Amazon EC2. Notez que, cette fois, vous pouvez voir toutes les informations et effectuer les actions que vous souhaitez. Si vous accédez à une autre console en tant qu'utilisateur, les messages d'accès refusé s'affichent. La raison en est que les politiques de ce didacticiel n'accordent l'accès qu'à Amazon EC2.

Ressources connexes

Pour plus d'informations, veuillez consulter les rubriques suivantes :

- Utilisation de l'authentification multifactorielle (MFA) dans AWS
- Activation des appareils MFA pour les utilisateurs de AWS
- Utilisation de dispositifs MFA avec votre page de connexion IAM

Identités IAM (utilisateurs, groupes d'utilisateurs et rôles)

> **Tip**
>
> Vous rencontrez des difficultés pour vous connecter à AWS ? Vérifiez que vous êtes sur la bonne page de connexion.
>
> - Pour vous connecter en tant que Utilisateur racine d'un compte AWS (propriétaire du compte), utilisez les informations d'identification que vous avez définies lors de la création du Compte AWS.
> - Pour vous connecter en tant qu'utilisateur IAM, utilisez les informations d'identification que votre administrateur de compte vous a données pour vous connecter à AWS.
> - Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l'URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l'utilisateur IAM Identity Center.
>
>   Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.
>
> Pour des didacticiels de connexion, consultez la section Comment se connecter à AWS dans le Guide de l'utilisateur Connexion à AWS .

> **Note**
>
> Si vous avez besoin d'assistance, n'utilisez pas le lien Commentaire indiqué sur cette page. Les commentaires que vous saisissez sont reçus par l'équipe de AWS documentation, et non par AWS le Support. Choisissez plutôt le lien Contactez-nous en haut de cette page. Vous y trouverez des liens vers des ressources qui vous aideront à obtenir l'assistance dont vous avez besoin.

L'utilisateur Utilisateur racine d'un compte AWS ou un utilisateur administratif du compte peut créer des identités IAM. Une identité IAM permet d'accéder à un Compte AWS. Un groupe d'utilisateurs IAM est un ensemble d'utilisateurs IAM gérés en tant qu'unité. Une identité IAM représente un

utilisateur humain ou une charge de travail programmatique, et peut être authentifiée puis autorisée à effectuer des actions dans AWS. Chaque identité IAM peut être associée à une ou plusieurs politiques. Les politiques déterminent les actions qu'un utilisateur, un rôle ou un membre d'un groupe d'utilisateurs peut effectuer, sur quelles AWS ressources et dans quelles conditions.

Compte AWS utilisateur root

Lorsque vous créez un Compte AWS, vous commencez par une identité de connexion unique qui donne un accès complet à toutes Services AWS les ressources du compte. Cette identité est appelée utilisateur Compte AWS root et est accessible en vous connectant avec l'adresse e-mail et le mot de passe que vous avez utilisés pour créer le compte.

> **Important**
>
> Il est vivement recommandé de ne pas utiliser l'utilisateur racine pour vos tâches quotidiennes. Protégez vos informations d'identification d'utilisateur racine et utilisez-les pour effectuer les tâches que seul l'utilisateur racine peut effectuer. Pour obtenir la liste complète des tâches qui nécessitent que vous vous connectiez en tant qu'utilisateur root, veuillez consulter Tâches nécessitant les informations d'identification de l'utilisateur root.

Utilisateurs IAM

Un utilisateur IAM est une identité au sein de vous Compte AWS qui possède des autorisations spécifiques pour une seule personne ou application. Dans la mesure du possible, nous vous recommandons, dans le cadre des bonnes pratiques, de vous appuyer sur des informations d'identification temporaires plutôt que de créer des utilisateurs IAM ayant des informations d'identification à long terme tels que les clés d'accès. Avant de créer des clés d'accès, passez en revue les alternatives aux clés d'accès à long terme. Si vous avez des cas d'utilisation spécifiques qui nécessitent des clés d'accès, nous vous recommandons de mettre à jour les clés d'accès en cas de besoin. Pour plus d'informations, consultez Mettre à jour les clés d'accès lorsque cela est nécessaire pour les cas d'utilisation nécessitant des informations d'identification à long terme. Pour ajouter des utilisateurs IAM à votre Compte AWS, consultezCréation d'un utilisateur IAM dans votre Compte AWS.

> **Note**
>
> Une bonne pratique en matière de sécurité consiste à fournir un accès à vos ressources par le biais de la fédération d'identité plutôt que de créer des utilisateurs IAM. Pour en savoir plus sur les situations spécifiques dans lesquelles un utilisateur IAM est nécessaire, veuillez consulter Quand créer un utilisateur IAM (au lieu d'un rôle).

Groupes d'utilisateurs IAM

Un groupe IAM est une identité qui concerne un ensemble d'utilisateurs IAM. Vous ne pouvez pas utiliser un groupe pour vous connecter. Vous pouvez utiliser les groupes pour spécifier des autorisations pour plusieurs utilisateurs à la fois. Les groupes permettent de gérer plus facilement les autorisations pour de grands ensembles d'utilisateurs. Par exemple, vous pouvez avoir un groupe appelé IAMPublishers et accorder à ce groupe les types d'autorisations dont les charges de travail de publication ont généralement besoin.

Rôles IAM

Un rôle IAM est une identité au sein de vous Compte AWS dotée d'autorisations spécifiques. S'il est comparable à un utilisateur IAM, il n'est toutefois pas associé à une personne déterminée. Vous pouvez assumer temporairement un rôle IAM dans le en AWS Management Console changeant de rôle. Vous pouvez assumer un rôle en appelant une opération d' AWS API AWS CLI ou en utilisant une URL personnalisée. Pour plus d'informations sur les méthodes d'utilisation des rôles, veuillez consulter la rubrique Utilisation de rôles IAM.

Les rôles IAM avec des informations d'identification temporaires sont utiles dans les cas suivants :

- Accès utilisateur fédéré – Pour attribuer des autorisations à une identité fédérée, vous créez un rôle et définissez des autorisations pour le rôle. Quand une identité externe s'authentifie, l'identité est associée au rôle et reçoit les autorisations qui sont définies par celui-ci. Pour obtenir des informations sur les rôles pour la fédération, consultez Création d'un rôle pour un fournisseur d'identité tiers (fédération) dans le Guide de l'utilisateur IAM. Si vous utilisez IAM Identity Center, vous configurez un jeu d'autorisations. IAM Identity Center met en corrélation le jeu d'autorisations avec un rôle dans IAM afin de contrôler à quoi vos identités peuvent accéder après leur authentification. Pour plus d'informations sur les jeux d'autorisations, consultez la rubrique Jeux d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .

- Autorisations d’utilisateur IAM temporaires : un rôle ou un utilisateur IAM peut endosser un rôle IAM pour profiter temporairement d’autorisations différentes pour une tâche spécifique.
- Accès intercompte : vous pouvez utiliser un rôle IAM pour permettre à un utilisateur (principal de confiance) d’un compte différent d’accéder aux ressources de votre compte. Les rôles constituent le principal moyen d’accorder l’accès intercompte. Toutefois, certains Services AWS vous permettent d’attacher une politique directement à une ressource (au lieu d’utiliser un rôle en tant que proxy). Pour connaître la différence entre les rôles et les politiques basées sur les ressources pour l'accès intercompte, veuillez consulter la rubrique Accès intercompte aux ressources dans IAM.
- Accès multiservices — Certains Services AWS utilisent des fonctionnalités dans d'autres Services AWS. Par exemple, lorsque vous effectuez un appel dans un service, il est courant que ce service exécute des applications dans Amazon EC2 ou stocke des objets dans Amazon S3. Un service peut le faire en utilisant les autorisations d’appel du principal, un rôle de service ou un rôle lié au service.
  - Sessions d'accès direct (FAS) : lorsque vous utilisez un utilisateur ou un rôle IAM pour effectuer des actions AWS, vous êtes considéré comme un mandant. Lorsque vous utilisez certains services, vous pouvez effectuer une action qui initie une autre action dans un autre service. FAS utilise les autorisations du principal appelant et Service AWS, associées Service AWS à la demande, pour adresser des demandes aux services en aval. Les demandes FAS ne sont effectuées que lorsqu'un service reçoit une demande qui nécessite des interactions avec d'autres personnes Services AWS ou des ressources pour être traitée. Dans ce cas, vous devez disposer d’autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur la politique relative à la transmission de demandes FAS, consultez Sessions de transmission d’accès.
  - Rôle de service : il s’agit d’un rôle IAM attribué à un service afin de réaliser des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer une fonction du service à partir d’IAM. Pour plus d’informations, consultez Création d’un rôle pour la délégation d’autorisations à un Service AWS dans le Guide de l’utilisateur IAM.
  - Rôle lié à un service — Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d’effectuer une action en votre nom. Les rôles liés à un service apparaissent dans votre Compte AWS répertoire et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.
- Applications exécutées sur Amazon EC2 : vous pouvez utiliser un rôle IAM pour gérer les informations d'identification temporaires pour les applications qui s'exécutent sur une instance EC2 et qui envoient des demandes d'API. AWS CLI AWS Cette solution est préférable au stockage

des clés d’accès au sein de l’instance EC2. Pour attribuer un AWS rôle à une instance EC2 et le mettre à la disposition de toutes ses applications, vous devez créer un profil d'instance attaché à l'instance. Un profil d’instance contient le rôle et permet aux programmes qui s’exécutent sur l’instance EC2 d’obtenir des informations d’identification temporaires. Pour plus d’informations, consultez Utilisation d’un rôle IAM pour accorder des autorisations à des applications s’exécutant sur des instances Amazon EC2 dans le Guide de l’utilisateur IAM.

Informations d'identification temporaires dans IAM

En guise de bonne pratique, utilisez des informations d'identification temporaires à la fois pour les utilisateurs humains et les charges de travail. Les informations d'identification temporaires sont utilisées principalement avec des rôles IAM, mais il existe également d'autres utilisations. Vous pouvez demander des informations d'identification temporaires ayant un ensemble d'autorisations plus restreintes que votre utilisateur IAM standard. Cela vous évite d’effectuer des tâches accidentellement qui ne sont pas autorisées par les informations d’identification plus restreintes. Les informations d'identification temporaires présentent l'avantage d'expirer automatiquement après une période définie. Vous contrôlez la durée de validité des informations d'identification.

Quand utiliser les utilisateurs IAM Identity Center ?

Nous recommandons à tous les utilisateurs humains d'utiliser IAM Identity Center pour accéder aux AWS ressources. IAM Identity Center permet d'améliorer considérablement l'accès aux AWS ressources en tant qu'utilisateur IAM. IAM Identity Center fournit :

- Un ensemble central d'identités et d'affectations
- Accès aux comptes de l'ensemble de AWS l'organisation
- Une connexion à votre fournisseur d'identité existant
- Des informations d'identification temporaires
- Une authentification multifactorielle (MFA)
- Une configuration MFA en libre-service pour le s utilisateurs finaux
- Une application administrative de l'utilisation de l'authentification MFA
- Authentification unique pour tous les droits Compte AWS

Pour plus d'informations, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Quand créer un utilisateur IAM (au lieu d'un rôle)

Nous vous recommandons de n'utiliser les utilisateurs IAM que pour les cas d'utilisation non pris en charge par les utilisateurs fédérés. Voici certaines des fonctionnalités les plus utilisées :

- Charges de travail qui ne peuvent pas utiliser de rôles IAM — Vous pouvez exécuter une charge de travail à partir d'un emplacement qui a besoin d'accéder à AWS. Dans certains cas, vous ne pouvez pas utiliser les rôles IAM pour fournir des informations d'identification temporaires, par exemple pour les WordPress plug-ins. Dans ces situations, utilisez les clés d'accès à long terme de l'utilisateur IAM pour cette charge de travail afin de vous authentifier auprès de AWS.
- AWS Clients tiers : si vous utilisez des outils qui ne prennent pas en charge l'accès avec IAM Identity Center, tels que des AWS clients tiers ou des fournisseurs qui ne sont pas hébergés sur le site AWS, utilisez les clés d'accès à long terme des utilisateurs IAM.
- AWS CodeCommit accès — Si vous avez l'habitude de CodeCommit stocker votre code, vous pouvez utiliser un utilisateur IAM doté de clés SSH ou d'informations d'identification spécifiques au service pour vous authentifier CodeCommit auprès de vos référentiels. Nous vous recommandons de procéder ainsi en plus de vous servir d'un utilisateur dans IAM Identity Center pour une authentification ordinaire. Les utilisateurs d'IAM Identity Center sont les membres de votre personnel qui ont besoin d'accéder à vos applications cloud Comptes AWS ou à celles de celles-ci. Pour permettre aux utilisateurs d'accéder à vos CodeCommit référentiels sans configurer les utilisateurs IAM, vous pouvez configurer l'git-remote-codecommitutilitaire. Pour plus d'informations sur IAM et CodeCommit, consultezUtilisation d'IAM avec CodeCommit : informations d'identification Git, clés SSH et AWS clés d'accès. Pour plus d'informations sur la configuration de l'git-remote-codecommitutilitaire, consultez la section Connexion aux AWS CodeCommit référentiels avec des informations d'identification rotatives dans le Guide de AWS CodeCommit l'utilisateur.
- Amazon Keyspaces (for Apache Cassandra) access (Accès Amazon Keyspaces (pour Apache Cassandra)) : dans une situation où vous n'êtes pas en mesure de vous servir des utilisateurs dans IAM Identity Center, par exemple à des fins de test de compatibilité avec Cassandra, vous pouvez utiliser un utilisateur IAM avec des informations d'identification spécifiques au service pour vous authentifier auprès d'Amazon Keyspaces. Les utilisateurs d'IAM Identity Center sont les membres de votre personnel qui ont besoin d'accéder à vos applications cloud Comptes AWS ou à celles de celles-ci. Vous pouvez également vous connecter à Amazon Keyspaces à l'aide d'informations d'identification temporaires. Pour de plus d'informations, veuillez consulter Utilisation d'informations d'identification temporaires pour se connecter à Amazon Keyspaces à l'aide d'un rôle IAM et du plugin Sigv4 dans le Guide du développeur Amazon Keyspaces (pour Apache Cassandra).

- Accès d’urgence : dans une situation où vous n’avez pas accès à votre fournisseur d’identité et où vous devez prendre des mesures sur votre Compte AWS. La création d'utilisateurs IAM bénéficiant d'un accès d'urgence peut faire partie de votre plan de résilience. Nous vous recommandons de veiller à ce que les informations d'identification des utilisateurs d'urgence soient étroitement contrôlées et sécurisées à l'aide de l'authentification multifactorielle (MFA).

Quand créer un rôle IAM (au lieu d'un utilisateur)

Créez un rôle IAM dans les cas suivants :

Vous créez une application qui s'exécute sur une instance Amazon Elastic Compute Cloud (Amazon EC2) et à laquelle cette application envoie des demandes. AWS

> Ne créez pas d'utilisateur IAM pour transmettre les informations d'identification de l'utilisateur à l'application ou intégrer les informations d'identification à l'application. À la place, créez un rôle IAM que vous attachez à l'instance EC2 pour transmettre aux applications qui s'exécutent sur l'instance des informations d'identification de sécurité temporaires. Lorsqu'une application utilise ces informations d'identification AWS, elle peut effectuer toutes les opérations autorisées par les politiques associées au rôle. Pour plus de détails, consultez Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2.

Vous créez une application qui s'exécute sur un téléphone mobile et réalise des appels vers AWS.

> Ne créez pas d'utilisateur IAM pour distribuer sa clé d'accès avec l'application. À la place, utilisez un fournisseur d'identité comme Login with Amazon, Amazon Cognito, Facebook ou Google pour authentifier les utilisateurs et les mapper à un rôle IAM. L'application utilise le rôle pour obtenir des informations d'identification de sécurité temporaires qui disposent des autorisations spécifiées par les politiques attachées au rôle. Pour plus d’informations, consultez les ressources suivantes :
>
> - Guide de l'utilisateur d'Amazon Cognito
> - Fédération OIDC

Les utilisateurs de votre entreprise sont authentifiés sur votre réseau d'entreprise et souhaitent pouvoir l'utiliser AWS sans avoir à se reconnecter, c'est-à-dire que vous souhaitez autoriser les utilisateurs à se fédérer dans. AWS

> Ne créez pas d'utilisateurs IAM. Configurez une relation de fédération entre votre système d'identité d'entreprise et AWS. Vous pouvez effectuer cette opération de deux façons :

- Si le système d'identité de votre entreprise est compatible avec SAML 2.0, vous pouvez établir un lien de confiance entre le système d'identité de votre entreprise et AWS. Pour plus d'informations, consultez Fédération SAML 2.0.
- Créez et utilisez un serveur proxy personnalisé qui traduit les identités des utilisateurs de l'entreprise en rôles IAM fournissant des informations d'identification AWS de sécurité temporaires. Pour plus d'informations, consultez Activation de l'accès à la AWS console par un courtier d'identité personnalisé.

Comparez les Utilisateur racine d'un compte AWS informations d'identification et les informations d'identification des utilisateurs IAM

L'utilisateur root est le propriétaire du compte et est créé lors de la Compte AWS création du. Les autres types d'utilisateurs, y compris les utilisateurs IAM, et AWS IAM Identity Center les utilisateurs sont créés par l'utilisateur root ou un administrateur du compte. Tous les AWS utilisateurs disposent d'identifiants de sécurité.

Informations d'identification de l'utilisateur root

Les informations d'identification du propriétaire du compte permettent un accès complet à toutes les ressources du compte. Vous ne pouvez pas utiliser les politiques IAM pour refuser l'accès aux ressources de manière explicite à l'utilisateur root. Vous ne pouvez utiliser une politique AWS Organizations de contrôle des services (SCP) que pour limiter les autorisations de l'utilisateur root d'un compte membre. C'est pourquoi nous vous recommandons de créer un utilisateur administratif dans IAM Identity Center à utiliser pour les AWS tâches quotidiennes. Protégez ensuite les informations d'identification de l'utilisateur root et ne les utilisez que pour effectuer les quelques tâches de gestion des comptes et des services qui nécessitent que vous vous connectiez en tant que tel. Pour la liste de ces tâches, veuillez consulter Tâches nécessitant les informations d'identification de l'utilisateur root. Pour savoir comment configurer un administrateur pour une utilisation quotidienne dans IAM Identity Center, veuillez consulter la rubrique Getting started dans le Guide de l'utilisateur IAM Identity Center.

Informations d'identification IAM

Un utilisateur IAM est une entité que vous créez et AWS qui représente la personne ou le service qui utilise l'utilisateur IAM pour interagir avec AWS les ressources. Ces utilisateurs sont des identités au sein de vous Compte AWS qui disposent d'autorisations personnalisées spécifiques. Par exemple,

vous pouvez créer des utilisateurs IAM et leur donner l'autorisation de créer un répertoire dans IAM Identity Center. Les utilisateurs IAM disposent d'informations d'identification à long terme qu'ils peuvent utiliser pour accéder à l' AWS AWS Management Console aide des API or ou par programmation. AWS CLI AWS Pour step-by-step obtenir des instructions sur la manière dont les utilisateurs IAM se connectent au AWS Management Console, voir Se connecter en AWS Management Console tant qu'utilisateur IAM dans le guide de l'utilisateur de AWS connexion.

En général, nous vous recommandons d'éviter de créer des utilisateurs IAM, car ils possèdent des informations d'identification à long terme, telles qu'un nom d'utilisateur et un mot de passe. Demandez plutôt aux utilisateurs humains d'utiliser des informations d'identification temporaires lors de l'accès AWS. Vous pouvez utiliser un fournisseur d'identité auquel vos utilisateurs humains fourniront un accès fédéré Comptes AWS en assumant des rôles IAM, qui fournissent des informations d'identification temporaires. Pour une gestion centralisée des accès, nous vous recommandons d'utiliser IAM Identity Center pour gérer l'accès à vos comptes et les autorisations au sein de ceux-ci. Vous pouvez gérer vos identités d'utilisateur avec IAM Identity Center ou gérer les autorisations d'accès pour les identités des utilisateurs dans IAM Identity Center à partir d'un fournisseur d'identité externe. Pour plus d'informations, veuillez consulter la rubrique What is IAM Identity Center dans le Guide de l'utilisateur IAM Identity Center.

Utilisateur racine d'un compte AWS

Lorsque vous créez un compte Amazon Web Services (AWS) pour la première fois, vous utilisez une identité de connexion unique qui donne un accès complet à tous les AWS services et ressources du compte. Cette identité est appelée utilisateur root du AWS compte et est accessible en vous connectant avec l'adresse e-mail et le mot de passe que vous avez utilisés pour créer le compte.

> **Important**
>
> Nous vous recommandons vivement de ne pas utiliser l'utilisateur root pour vos tâches quotidiennes et de suivre les bonnes pratiques de l'utilisateur root pour votre Compte AWS. Protégez vos informations d'identification d'utilisateur racine et utilisez-les pour effectuer les tâches que seul l'utilisateur racine peut effectuer. Pour obtenir la liste complète des tâches qui nécessitent que vous vous connectiez en tant qu'utilisateur root, veuillez consulter Tâches nécessitant les informations d'identification de l'utilisateur root.

Les rubriques suivantes détaillent les tâches de gestion associées à l'utilisateur root.

Tâches

- Activez l'authentification multifactorielle pour votre Utilisateur racine d'un compte AWS (console)
- Modifiez le mot de passe du Utilisateur racine d'un compte AWS
- Réinitialisation d'un mot de passe d'utilisateur racine perdu ou oublié
- Création de clés d'accès pour l'utilisateur racine
- Suppression des clés d'accès pour l'utilisateur racine
- Tâches nécessitant les informations d'identification de l'utilisateur root
- Résolution de problèmes relatifs à l'utilisateur root
- Informations connexes

Activez l'authentification multifactorielle pour votre Utilisateur racine d'un compte AWS (console)

L'authentification multifactorielle (MFA) est un mécanisme simple et efficace pour renforcer votre sécurité. Le premier facteur, votre mot de passe, est un secret que vous mémorisez, également appelé facteur de connaissance. Les autres facteurs peuvent être des facteurs liés à la possession (ce que vous possédez, comme une clé de sécurité) ou des facteurs inhérents (ce que vous êtes, comme un scan biométrique). Pour une sécurité accrue, nous vous recommandons vivement de configurer l'authentification multifactorielle (MFA) afin de protéger AWS vos ressources.

Vous pouvez activer le MFA pour les utilisateurs Utilisateur racine d'un compte AWS et IAM. Lorsque vous activez le MFA pour l'utilisateur root, cela n'affecte que les informations d'identification de l'utilisateur root. Pour plus d'informations sur la façon d'activer l'authentification multifacteur pour vos utilisateurs IAM, consultez la section Activation des appareils MFA pour les utilisateurs IAM dans. AWS

Avant d'activer le MFA pour votre utilisateur root, vérifiez et mettez à jour les paramètres de votre compte et vos coordonnées pour vous assurer que vous avez accès à l'e-mail et au numéro de téléphone. Si votre dispositif MFA est perdu, volé ou ne fonctionne pas, vous pouvez toujours vous connecter en tant qu'utilisateur racine en vérifiant votre identité à l'aide de cet e-mail et de ce numéro de téléphone. Pour en savoir plus sur la connexion à l'aide d'autres facteurs d'authentification, consultez Que faire si un dispositif MFA est perdu ou cesse de fonctionner ?. Pour désactiver cette fonction, contactez AWS Support.

Rubriques

- Types MFA disponibles pour un utilisateur root
- Activer une clé d'accès ou une clé de sécurité pour la Utilisateur racine d'un compte AWS (console)
- Activation d'un dispositif MFA virtuel pour votre Utilisateur racine d'un compte AWS (console)
- Activer un jeton TOTP matériel pour Utilisateur racine d'un compte AWS (console)

Types MFA disponibles pour un utilisateur root

AWS prend en charge les types MFA suivants pour votre utilisateur root : clés d'accès et clés de sécurité, applications d'authentification virtuelle et jetons TOTP matériels.

Clés d'accès et clés de sécurité

AWS Identity and Access Management prend en charge les clés d'accès et les clés de sécurité pour la MFA. Basées sur les normes FIDO, les clés d'accès utilisent la cryptographie à clé publique pour fournir une authentification solide, résistante au hameçonnage et plus sécurisée que les mots de passe. AWS prend en charge deux types de clés d'accès : les clés d'accès liées à l'appareil (clés de sécurité) et les clés d'accès synchronisées.

- Clés de sécurité : il s'agit de périphériques physiques YubiKey, tels que a, utilisés comme deuxième facteur d'authentification.
- Clés d'accès synchronisées : elles utilisent des gestionnaires d'identifiants provenant de fournisseurs tels que Google, Apple, Microsoft et de services tiers tels que 1Password, Dashlane et Bitwarden comme deuxième facteur.

Vous pouvez utiliser des authentificateurs biométriques intégrés, tels que Touch ID sur Apple MacBooks et la reconnaissance faciale Windows Hello sur PC, pour déverrouiller votre gestionnaire d'identifiants et vous y connecter. AWS Les clés d'accès sont créées avec le fournisseur de votre choix à l'aide de votre empreinte digitale, de votre visage ou du code PIN de votre appareil. Vous pouvez synchroniser les clés d'accès sur tous vos appareils pour faciliter les connexions et améliorer la convivialité AWS et la récupérabilité.

La FIDO Alliance tient à jour une liste de tous les produits certifiés FIDO qui sont compatibles avec les spécifications FIDO. Une clé d'accès ou une clé de sécurité unique prend en charge plusieurs comptes utilisateur root et utilisateurs IAM. Pour plus d'informations sur l'activation des clés d'accès et des clés de sécurité, consultezActiver une clé d'accès ou une clé de sécurité pour la Utilisateur racine d'un compte AWS (console).

Applications d'authentification virtuelle

Une application d'authentification virtuelle s'exécute sur un téléphone ou un autre appareil et émule un appareil physique. Les applications d'authentification virtuelle mettent en œuvre l'algorithme TOTP (mot de passe unique à durée limitée) et prennent en charge plusieurs jetons sur un seul dispositif. L'utilisateur doit saisir un code valide sur l'appareil lorsqu'il y est invité lors de la connexion. Chaque jeton attribué à un utilisateur doit être unique. Un utilisateur ne peut pas saisir de code à partir du jeton d'un autre utilisateur pour s'authentifier.

Nous vous recommandons d'utiliser un dispositif MFA virtuel pendant l'attente de l'approbation d'achat du matériel ou pendant que vous attendez de recevoir votre matériel. Pour obtenir la liste de quelques applications prises en charge que vous pouvez utiliser comme appareils MFA virtuels, consultez la section Multi-Factor Authentication (MFA). Pour obtenir des instructions sur la configuration d'un périphérique MFA virtuel avec AWS, voir. Activation d'un dispositif MFA virtuel pour votre Utilisateur racine d'un compte AWS (console)

Tokens TOTP matériels

Un périphérique matériel génère un code numérique à six chiffres basé sur l'algorithme TOTP (mot de passe à usage unique basé sur le temps). L'utilisateur doit saisir un code valide à partir du dispositif sur une deuxième page web lors de la connexion. Chaque dispositif MFA attribué à un utilisateur doit être unique. Un utilisateur ne peut pas saisir un code à partir du périphérique d'un autre utilisateur pour s'authentifier. Pour plus d'informations sur les dispositifs matériels MFA pris en charge, consultez la section Multi-Factor Authentication (MFA). Pour obtenir des instructions sur la configuration d'un jeton TOTP matériel avec AWS, voirActiver un jeton TOTP matériel pour Utilisateur racine d'un compte AWS (console).

Si vous souhaitez utiliser un périphérique MFA physique, nous vous recommandons d'utiliser les clés de sécurité FIDO comme alternative aux périphériques TOTP matériels. Les clés de sécurité FIDO offrent les avantages de ne pas nécessiter de batterie, de résister au phishing et de prendre en charge plusieurs utilisateurs root et IAM sur un seul appareil pour une sécurité renforcée.

Activer une clé d'accès ou une clé de sécurité pour la Utilisateur racine d'un compte AWS (console)

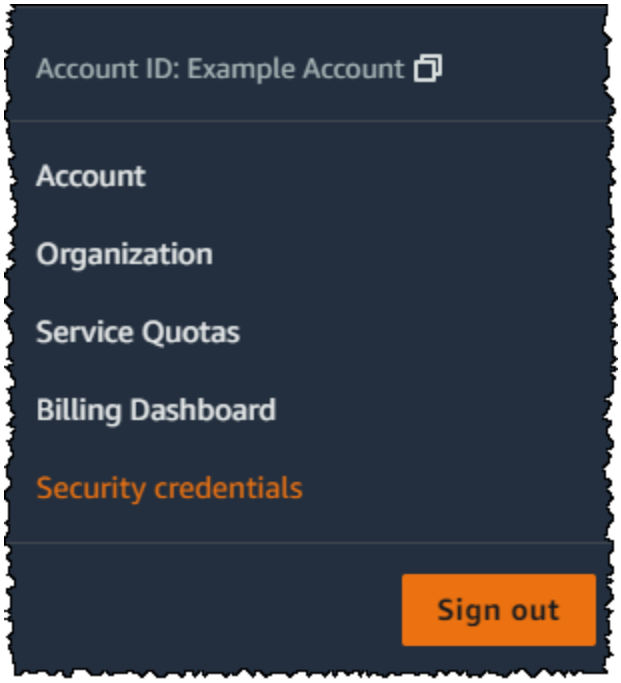
Vous pouvez configurer et activer une clé d'accès pour votre utilisateur root AWS Management Console uniquement, et non depuis l' AWS API AWS CLI or.

Pour activer une clé d'accès ou une clé de sécurité pour votre utilisateur root (console)

1. Connectez-vous à la console IAM en tant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   > **Note**
   >
   > En tant qu'utilisateur root, vous ne pouvez pas vous connecter à la page Se connecter en tant qu'utilisateur IAM. Si la page Se connecter en tant qu'utilisateur IAM s'affiche, choisissez Se connecter à l'aide de l'adresse e-mail de l'utilisateur root en bas de la page. Pour obtenir de l'aide pour vous connecter en tant qu'utilisateur root, consultez la section Connexion en AWS Management Console tant qu'utilisateur root dans le Guide de Connexion à AWS l'utilisateur.

2. À droite de la barre de navigation, choisissez le nom de votre compte, et cliquez sur Security credentials (Informations d'identification de sécurité). Au besoin, choisissez Continue to Security credentials (Passer aux informations d'identification de sécurité).



3. Sur la page Mes identifiants de sécurité de votre utilisateur root, sous Authentification à facteurs multiples (MFA), choisissez Attribuer un appareil MFA.
4. Sur la page du nom de l'appareil MFA, entrez un nom d'appareil, choisissez Clé d'accès ou Clé de sécurité, puis choisissez Suivant.

5. Dans Configurer l'appareil, configurez votre clé d'accès. Créez une clé d'accès avec des données biométriques telles que votre visage ou votre empreinte digitale, avec le code PIN d'un appareil, ou en insérant la clé de sécurité FIDO dans le port USB de votre ordinateur et en la touchant.
6. Suivez les instructions de votre navigateur pour choisir un fournisseur de clé d'accès ou l'endroit où vous souhaitez stocker votre clé d'accès afin de l'utiliser sur tous vos appareils.
7. Choisissez Continuer.

Vous avez maintenant enregistré votre clé d'accès pour l'utiliser avec AWS. La prochaine fois que vous utiliserez vos informations d'identification d'utilisateur root pour vous connecter, vous devrez vous authentifier avec votre clé d'accès pour terminer le processus de connexion.

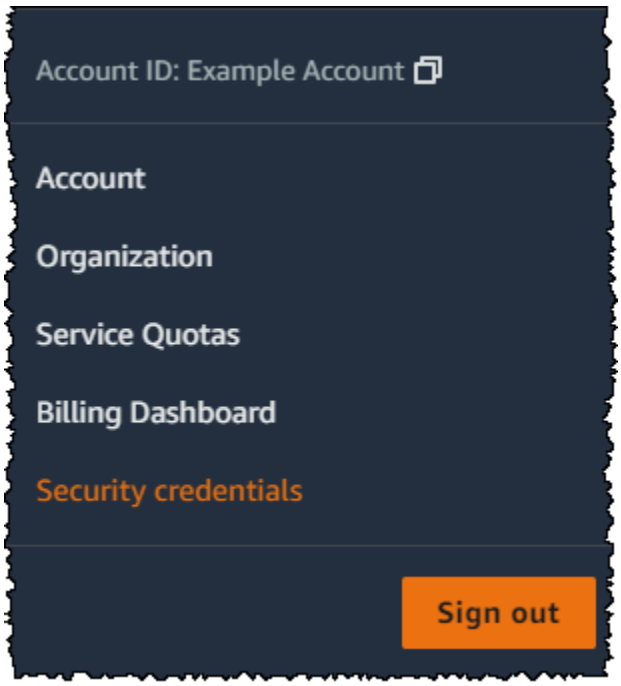
Pour obtenir de l'aide pour résoudre les problèmes liés à votre clé de sécurité FIDO, consultezRésolution des problèmes liés aux clés de sécurité FIDO.

Activation d'un dispositif MFA virtuel pour votre Utilisateur racine d'un compte AWS (console)

Vous pouvez utiliser le AWS Management Console pour configurer et activer un périphérique MFA virtuel pour votre utilisateur root. Pour activer les appareils MFA pour le Compte AWS, vous devez être connecté à l' AWS aide de vos informations d'identification d'utilisateur root.

Pour configurer et activer un dispositif MFA virtuel à utiliser avec votre utilisateur racine (console)

1. Connectez-vous au AWS Management Console.
2. À droite de la barre de navigation, choisissez le nom de votre compte, et cliquez sur Security credentials (Informations d'identification de sécurité). Au besoin, choisissez Continue to Security credentials (Passer aux informations d'identification de sécurité).



3. Dans la section Multi-Factor Authentication (MFA) (Authentification multifactorielle (MFA)), sélectionnez Assign MFA device (Attribuer un dispositif MFA).
4. Dans l'assistant, saisissez un nom dans le champ Nom du dispositif, sélectionnez Application Authenticator, puis cliquez sur Suivant.

   IAM génère et affiche les informations de configuration du dispositif MFA virtuel, notamment un graphique de code QR. Le graphique est une représentation de la clé de configuration secrète que l'on peut saisir manuellement sur des dispositifs qui ne prennent pas en charge les codes QR.
5. Ouvrez l'application MFA virtuelle sur l'appareil.

   Si l'application MFA virtuelle prend en charge plusieurs comptes ou plusieurs dispositifs MFA virtuels, choisissez l'option permettant de créer un compte ou un dispositif MFA virtuel.
6. La manière la plus simple de configurer l'application consiste à utiliser l'application pour analyser le code QR. Si vous ne pouvez pas analyser le code, vous pouvez saisir les informations de configuration manuellement. Le code QR et la clé de configuration secrète générés par IAM sont liés à votre compte Compte AWS et ne peuvent pas être utilisés avec un autre compte. En revanche, ils peuvent être réutilisés pour configurer un nouveau dispositif MFA pour votre compte si vous perdez l'accès au dispositif MFA d'origine.

   - Pour utiliser le code QR pour configurer le dispositif MFA virtuel, dans l'assistant, choisissez Show QR code (Afficher le code QR). Ensuite, suivez les instructions de l'application pour

scanner le code. Par exemple, vous pouvez avoir besoin de choisir l'icône de caméra ou une commande similaire à Scan account barcode (Analyser le code-barres du compte), puis d'utiliser la caméra du périphérique pour analyser le code QR.

- Dans l'assistant Set up device (Configurer le dispositif), sélectionnez Show secret key (Afficher la clé secrète), puis saisissez la clé secrète dans votre application MFA.

> **Important**
>
> Faites une sauvegarde sécurisée du code QR ou de la clé de configuration secrète ou assurez-vous d'activer plusieurs dispositifs MFA pour votre compte. Vous pouvez enregistrer jusqu'à huit appareils MFA de n'importe quelle combinaison des types de MFA actuellement pris en charge auprès de vos Utilisateur racine d'un compte AWS utilisateurs et de ceux d'IAM. Un dispositif MFA virtuel peut devenir indisponible, par exemple, si vous perdez le smartphone hébergeant le dispositif MFA virtuel. Si cela se produit et que vous ne parvenez pas à vous connecter à votre compte sans autre dispositif MFA associé à l'utilisateur ou même d'après Récupération d'un dispositif MFA d'utilisateur racine, vous ne pourrez pas vous connecter à votre compte et vous devrez contacter le service client pour supprimer la protection MFA du compte.

Le périphérique commence à générer des numéros à six chiffres.

7. Dans l'assistant, dans la zone MFA Code 1 (Code MFA 1), saisissez le mot de passe unique qui s'affiche actuellement sur le dispositif MFA virtuel. Attendez jusqu'à 30 secondes pour que le dispositif génère un nouveau mot de passe unique. Saisissez ensuite le second mot de passe unique dans la zone MFA Code 2 (Code MFA 2). Choisissez Add MFA (Ajouter un dispositif MFA).

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré le code. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, le dispositif MFA s'associe avec succès à l'utilisateur mais est désynchronisé. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période. Dans ce cas, vous pouvez resynchroniser le dispositif.

L'appareil est prêt à être utilisé avec AWS. Pour plus d'informations sur l'utilisation de l'authentification MFA avec l'interface AWS Management Console, veuillez consulter Utilisation de dispositifs MFA avec votre page de connexion IAM.

Activer un jeton TOTP matériel pour Utilisateur racine d'un compte AWS (console)

Vous pouvez configurer et activer un dispositif MFA physique pour votre utilisateur root AWS Management Console uniquement, et non à partir de l'API AWS CLI or AWS .

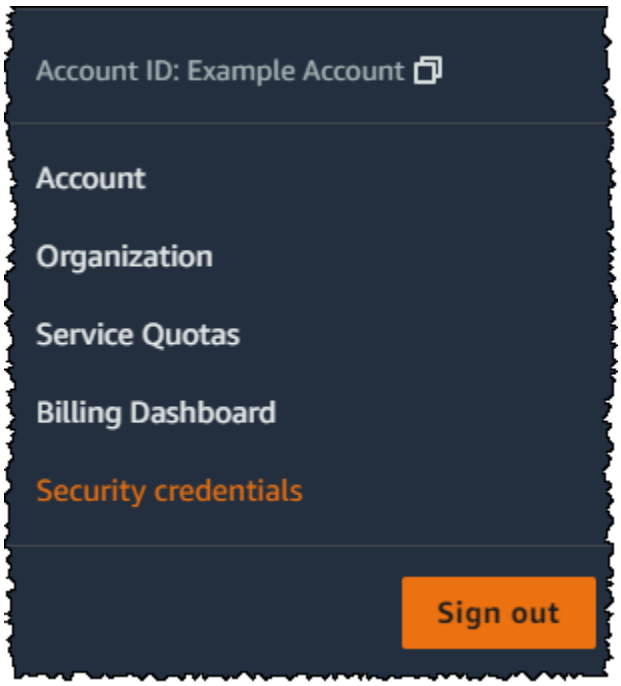
> **Note**
>
> Il se peut que vous remarquiez des textes différents, tels que se connecter à l'aide de MFA et dépanner votre dispositif d'authentification. Toutefois, les mêmes fonctions sont fournies. Dans ces deux cas, si vous ne pouvez pas vérifier l'adresse e-mail et le numéro de téléphone de votre compte à l'aide d'autres facteurs d'authentification, contactez AWS Support pour désactiver votre paramètre MFA.

Pour activer le dispositif MFA pour votre utilisateur racine (console)

1. Connectez-vous à la console IAM en tant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   > **Note**
   >
   > En tant qu'utilisateur root, vous ne pouvez pas vous connecter à la page Se connecter en tant qu'utilisateur IAM. Si la page Se connecter en tant qu'utilisateur IAM s'affiche, choisissez Se connecter à l'aide de l'adresse e-mail de l'utilisateur root en bas de la page. Pour obtenir de l'aide pour vous connecter en tant qu'utilisateur root, consultez la section Connexion en AWS Management Console tant qu'utilisateur root dans le Guide de Connexion à AWS l'utilisateur.

2. À droite de la barre de navigation, sélectionnez le nom de votre compte, puis Security Credentials (Informations d'identification de sécurité). Au besoin, choisissez Continue to Security credentials (Passer aux informations d'identification de sécurité).



3. Développez la section Multi-factor authentication (MFA) (authentification multifactorielle (MFA)).
4. Choisissez Assign MFA device (Attribuer un dispositif MFA).
5. Dans l'assistant, tapez le nom du dispositif, choisissez Hardware TOTP token (Jeton TOTP matériel), puis Next (Suivant).
6. Dans la zone Numéro de série, saisissez le numéro de série qui se trouve à l'arrière du dispositif MFA.
7. Dans la zone MFA code 1, saisissez le code à six chiffres qui s'affichent sur le dispositif MFA. Vous devrez peut-être appuyer sur le bouton situé à l'avant du périphérique pour afficher le numéro.

8. Attendez 30 secondes que le périphérique actualise le code, puis saisissez la nouvelle série de six chiffres dans la zone MFA code 2. Vous devrez peut-être appuyer à nouveau sur le bouton situé à l'avant du périphérique pour afficher le second numéro.
9. Choisissez Add MFA (Ajouter un dispositif MFA). Le dispositif MFA est à présent associé au Compte AWS.

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes d'authentification. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, l'dispositif MFA s'associe avec succès à l'utilisateur mais se désynchronise. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période. Dans ce cas, vous pouvez resynchroniser le dispositif.

La prochaine fois que vous utiliserez les informations d'identification d'utilisateur racine, vous devrez saisir un code du dispositif MFA.

Modifiez le mot de passe du Utilisateur racine d'un compte AWS

Vous pouvez modifier l'adresse e-mail et le mot de passe à partir de la page Informations d'identification ou de la page Compte. Vous pouvez également choisir Mot de passe oublié ? sur la page de AWS connexion pour réinitialiser votre mot de passe.

Pour modifier le mot de passe de l'utilisateur root, vous devez vous connecter en tant qu'utilisateur Utilisateur racine d'un compte AWS et non en tant qu'utilisateur IAM. Pour savoir comment réinitialiser un mot de passe d'utilisateur racine oublié, veuillez consulter Réinitialisation d'un mot de passe d'utilisateur racine perdu ou oublié.

Pour protéger votre mot de passe, il est important de respecter les bonnes pratiques suivantes :

- Modifiez régulièrement votre mot de passe.
- Gardez votre mot de passe confidentiel, car toute personne qui le connaît peut accéder à votre compte.
- Utilisez un mot de passe différent AWS de celui que vous utilisez sur d'autres sites.
- Evitez les mots de passe trop faciles à deviner, notamment : `secret`, `password`, `amazon` ou `123456`. Évitez également les mots du dictionnaire, votre nom, votre adresse électronique ou d'autres informations personnelles que quelqu'un pourrait facilement obtenir.

AWS Management Console

Pour changer le mot de passe de l'utilisateur racine

> **Autorisations minimales**
>
> Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :
>
> - Vous devez vous connecter en tant qu'utilisateur Compte AWS root, ce qui ne nécessite aucune autorisation AWS Identity and Access Management (IAM) supplémentaire. Vous ne pouvez pas effectuer ces étapes en tant qu'utilisateur ou rôle IAM.

1. Utilisez votre Compte AWS adresse e-mail et votre mot de passe pour vous connecter en AWS Management Consoletant que votre Utilisateur racine d'un compte AWS.
2. Dans le coin supérieur droit de la console, choisissez votre nom ou votre numéro de compte, puis choisissez Compte.
3. Sur la page Compte, en regard de Paramètres du compte, choisissez Modifier. Pour des raisons de sécurité, vous êtes invité à vous réauthentifier.

   > **Note**
   >
   > Si l'option Modifier ne s'affiche pas, il est probable que vous ne soyez pas connecté en tant qu'utilisateur root de votre compte. Vous ne pouvez pas modifier les paramètres du compte lorsque vous êtes connecté en tant qu'utilisateur ou rôle IAM.

4. Sur la page Mettre à jour les paramètres du compte, sous Mot de passe, sélectionnez Modifier.
5. Sur la page Mettre à jour votre mot de passe, remplissez les champs Mot de passe actuel, Nouveau mot de passe et Confirmer le nouveau mot de passe.

   > **Important**
   >
   > Assurez-vous de choisir un mot de passe fort. Bien que vous puissiez définir une politique de mot de passe de compte pour les utilisateurs IAM, celle-ci ne s'applique pas à votre utilisateur root.

AWS nécessite que votre mot de passe remplisse les conditions suivantes :

- Avoir un minimum de 8 caractères et un maximum de 128 caractères
- Inclure au minimum trois des types de caractères suivants : majuscules, minuscules, chiffres, et les symboles ! @ # $ % ^ & * () <> [] {} | _ + - =
- Il ne doit pas être identique à votre Compte AWS nom ou à votre adresse e-mail.

> **Note**
>
> AWS apporte des améliorations au processus de connexion. L'une de ces améliorations consiste à appliquer une politique de mot de passe plus sécurisée pour votre compte. Si AWS a mis à niveau votre compte, vous êtes tenu de respecter la politique de mot de passe ci-dessus. Si vous AWS n'avez pas encore mis à jour votre compte, cette politique AWS n'est pas encore appliquée. Cependant, nous vous recommandons vivement de suivre ses directives pour un mot de passe plus sûr.

6. Sélectionnez Enregistrer les modifications.

AWS CLI or AWS SDK

Cette tâche n'est pas prise en charge dans AWS CLI ou par une opération d'API provenant de l'un des AWS SDK. Vous ne pouvez effectuer cette tâche qu'à l'aide du AWS Management Console.

Réinitialisation d'un mot de passe d'utilisateur racine perdu ou oublié

Lorsque vous avez créé votre Compte AWS, vous avez fourni une adresse e-mail et un mot de passe. Voici vos Utilisateur racine d'un compte AWS informations d'identification. Si vous oubliez votre mot de passe utilisateur racine, vous pouvez le réinitialiser à partir de la AWS Management Console.

Pour réinitialiser votre mot de passe utilisateur racine :

1. Utilisez votre adresse Compte AWS e-mail pour commencer à vous connecter en AWS Management Consoletant qu'utilisateur root, puis choisissez Next.

> **Note**
>
> Si vous vous êtes connecté à AWS Management Console avec des informations d'identification d'utilisateur IAM, vous devez vous déconnecter pour pouvoir réinitialiser le mot de passe utilisateur racine. Si la page de connexion utilisateur IAM spécifique au compte s'affiche, choisissez Identifiez-vous à l'aide de vos informations de connexion au compte racine située près du bas de la page. Si nécessaire, fournissez l'adresse e-mail de votre compte et choisissez Next (Suivant) pour accéder à la page Root user sign in (Connexion de l'utilisateur racine).

2. Choisissez Forgot your password? (Mot de passe oublié ?).

> **Note**
>
> Si vous êtes un utilisateur IAM, cette option n'est pas disponible. L'option Mot de passe oublié ? n'est disponible que pour le compte utilisateur root. Les utilisateurs d'IAM doivent demander à leur administrateur de réinitialiser un mot de passe oublié. Pour plus d'informations, voir J'ai oublié le mot de passe utilisateur IAM associé à mon AWS compte. Si vous vous connectez via le Portail d'accès AWS, voir Réinitialisation de votre mot de passe utilisateur IAM Identity Center.

3. Indiquez l'adresse e-mail associée au compte. Indiquez ensuite le texte CAPTCHA et choisissez Continue (Continuer).
4. Vérifiez que l'e-mail qui vous est associé ne contient Compte AWS pas de message provenant d'Amazon Web Services. L'e-mail peut provenir d'une adresse se terminant par dans `@verify.signin.aws`. Suivez les instructions de l'e-mail. Si vous ne voyez pas l'e-mail dans votre compte, vérifiez le dossier des courriers indésirables. Si vous n'avez plus accès à l'e-mail, consultez la section Je n'ai pas accès à l'e-mail associé à mon AWS compte dans le guide de Connexion à AWS l'utilisateur.

Création de clés d'accès pour l'utilisateur racine

> **Warning**
>
> Nous vous recommandons vivement de ne pas créer de paires de clés d'accès pour votre utilisateur root. Étant donné que seules quelques tâches nécessitent l'utilisateur root et que

vous les effectuez généralement rarement, nous vous recommandons de vous connecter au AWS Management Console pour effectuer les tâches de l'utilisateur root. Avant de créer des clés d'accès, passez en revue les alternatives aux clés d'accès à long terme.

Bien que cela ne soit pas recommandé, vous pouvez créer des clés d'accès pour votre utilisateur root afin de pouvoir exécuter des commandes dans le AWS Command Line Interface (AWS CLI) ou utiliser des opérations d'API depuis l'un des AWS SDK à l'aide des informations d'identification de l'utilisateur root. Lorsque vous créez des clés d'accès, vous créez l'ID de clé d'accès et la clé d'accès secrète sous forme d'ensemble. Lors de la création de la clé AWS d'accès, vous avez la possibilité de consulter et de télécharger la partie clé d'accès secrète de la clé d'accès. Si vous ne la téléchargez pas ou si vous la perdez, vous pouvez supprimez la clé d'accès, puis en créer une nouvelle. Vous pouvez créer des clés d'accès utilisateur root à l'aide de la console ou de AWS l'API. AWS CLI

Une clé d'accès nouvellement créée a le statut active, ce qui signifie que vous pouvez l'utiliser pour les appels de CLI et d'API. Vous pouvez attribuer jusqu'à deux clés d'accès à l'utilisateur root.

Les clés d'accès qui ne sont pas utilisées doivent être désactivées. Lorsqu'une clé d'accès est inactive, vous ne pouvez pas l'utiliser pour les appels d'API. Les clés inactives comptent toujours dans votre limite. Vous pouvez créer ou supprimer une clé d'accès à tout moment. Toutefois, la suppression d'une clé d'accès est définitive et vous ne pourrez plus la récupérer.

AWS Management Console

Pour créer une clé d'accès pour Utilisateur racine d'un compte AWS

> **Autorisations minimales**
>
> Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :
>
> - Vous devez vous connecter en tant qu'utilisateur Compte AWS root, ce qui ne nécessite aucune autorisation AWS Identity and Access Management (IAM) supplémentaire. Vous ne pouvez pas effectuer ces étapes en tant qu'utilisateur ou rôle IAM.

1. Utilisez votre Compte AWS adresse e-mail et votre mot de passe pour vous connecter au Getting Started avec AWS Management Console le Utilisateur racine d'un compte AWS.

2. Dans le coin supérieur droit de la console, choisissez votre nom ou votre numéro de compte, puis sélectionnez Informations d'identification de sécurité.
3. Dans la section Clés d'accès, choisissez Créer une clé d'accès. Si cette option n'est pas disponible, cela signifie que vous avez déjà atteint le nombre maximum de clés d'accès. Vous devez supprimer l'une des clés d'accès existantes avant de pouvoir créer une clé. Pour plus d'informations, veuillez consulter Quotas d'objets IAM.
4. Sur la page Alternatives aux clés d'accès d'utilisateur root, passez en revue les recommandations de sécurité. Pour continuer, cochez la case, puis choisissez Créer une clé d'accès.
5. Sur la page Récupérer la clé d'accès, votre ID de clé d'accès est affiché.
6. Sous Clé d'accès secrète, choisissez Afficher puis copiez l'ID de la clé d'accès et la clé secrète dans la fenêtre de votre navigateur et collez-les dans un endroit sûr. Vous pouvez également choisir l'option Télécharger un fichier .csv qui téléchargera un fichier nommé `rootkey.csv` contenant l'ID de la clé d'accès et la clé secrète. Enregistrez le fichier à un endroit sûr.
7. Sélectionnez Exécuté. Lorsque vous n'avez plus besoin d'utiliser la clé d'accès, nous vous recommandons de la supprimer, ou au moins d'envisager de la désactiver afin que personne ne puisse en abuser.

AWS CLI & SDKs

Pour créer une clé d'accès pour l'utilisateur root

> **Note**
>
> Pour exécuter la commande ou l'opération d'API suivante en tant qu'utilisateur root, vous devez déjà disposer d'une paire de clés d'accès active. Si vous n'avez aucune clé d'accès, créez la première clé d'accès à l'aide de l' AWS Management Console. Vous pouvez ensuite utiliser les informations d'identification de cette première clé d'accès avec le AWS CLI pour créer la deuxième clé d'accès ou pour supprimer une clé d'accès.

- AWS CLI: cime create-access-key

  Example

```
$ aws iam create-access-key
```

```
{
    "AccessKey": {
        "UserName": "MyUserName",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "Status": "Active",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "CreateDate": "2021-04-08T19:30:16+00:00"
    }
}
```

- AWS API : CreateAccessKeydans la référence d'API IAM.

Suppression des clés d'accès pour l'utilisateur racine

Vous pouvez utiliser l' AWS Management Console API AWS CLI ou l' AWS API pour supprimer les clés d'accès de l'utilisateur root.

AWS Management Console

Pour supprimer une clé d'accès de l'utilisateur root

> **Autorisations minimales**
>
> Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :
>
> - Vous devez vous connecter en tant qu'utilisateur Compte AWS root, ce qui ne nécessite aucune autorisation AWS Identity and Access Management (IAM) supplémentaire. Vous ne pouvez pas effectuer ces étapes en tant qu'utilisateur ou rôle IAM.

1. Utilisez votre Compte AWS adresse e-mail et votre mot de passe pour vous connecter au Getting Started avec AWS Management Console le Utilisateur racine d'un compte AWS.
2. Dans le coin supérieur droit de la console, choisissez votre nom ou votre numéro de compte, puis sélectionnez Informations d'identification de sécurité.
3. Dans la section Clés d'accès, sélectionnez la clé d'accès que vous souhaitez supprimer, puis sous Actions, choisissez Supprimer.

> **Note**
>
> Vous pouvez également Désactiver une clé d'accès au lieu de la supprimer définitivement. Vous pourrez ainsi la réutiliser ultérieurement sans avoir à modifier l'ID de clé ou la clé secrète. Lorsque la clé est inactive, toute tentative de l'utiliser dans les demandes adressées à l' AWS API échoue et l'accès est refusé en cas d'erreur.

4. Dans la boîte de dialogue Supprimer <access key ID>, choisissez Désactiver, saisissez l'ID de la clé d'accès pour confirmer que vous souhaitez la supprimer, puis choisissez Supprimer.

AWS CLI & SDKs

Pour supprimer une clé d'accès de l'utilisateur root

> **Autorisations minimales**
>
> Pour effectuer les étapes suivantes, vous devez au moins disposer des autorisations IAM suivantes :
>
> - Vous devez vous connecter en tant qu'utilisateur Compte AWS root, ce qui ne nécessite aucune autorisation AWS Identity and Access Management (IAM) supplémentaire. Vous ne pouvez pas effectuer ces étapes en tant qu'utilisateur ou rôle IAM.

- AWS CLI: cime delete-access-key

  Example

  ```
  $ aws iam delete-access-key \
      --access-key-id AKIAIOSFODNN7EXAMPLE
  ```

  Cette commande ne produit aucune sortie lorsqu'elle réussit.

- AWS API : DeleteAccessKey

Tâches nécessitant les informations d'identification de l'utilisateur root

> **Important**
>
> Vous rencontrez des difficultés pour vous connecter à AWS ? Assurez-vous que vous êtes sur la bonne page de connexion AWS pour votre type d'utilisateur. Si vous êtes le Utilisateur racine d'un compte AWS (propriétaire du compte), vous pouvez vous connecter à AWS l'aide des informations d'identification que vous avez définies lors de la création du Compte AWS. Si vous êtes un utilisateur IAM, votre administrateur de compte peut vous fournir les informations d'identification que vous pouvez utiliser pour vous connecter à AWS. Si vous avez besoin d'assistance, n'utilisez pas le lien de commentaires sur cette page, car le formulaire est reçu par l'équipe de AWS documentation, non AWS Support. Sur la page Contactez-nous, sélectionnez Impossible de vous connecter à votre AWS compte, puis choisissez l'une des options d'assistance disponibles.

Nous vous recommandons de configurer un utilisateur administratif AWS IAM Identity Center pour effectuer les tâches quotidiennes et accéder aux AWS ressources. Toutefois, vous ne pouvez effectuer les tâches énumérées ci-dessous que si vous vous connectez en tant qu'utilisateur root d'un compte.

Tâches de gestion de compte

- Modifier les paramètres de votre compte. Cela inclut le nom du compte, l'adresse e-mail, ainsi que le mot de passe et les clés d'accès de l'utilisateur root. Les autres paramètres du compte, tels que les informations de contact, la devise de paiement préférée Régions AWS, ne nécessitent pas d'informations d'identification d'utilisateur root.
- Restaurer les autorisations de l'utilisateur IAM. Si un utilisateur IAM révoque accidentellement ses propres autorisations, vous pouvez vous connecter en tant qu'utilisateur root pour modifier les politiques et restaurer ces autorisations.
- Fermez votre Compte AWS.

  Pour plus d’informations, consultez les rubriques suivantes :

  - Comment puis-je en attribuer la propriété Compte AWS à une autre entité ? .
  - Comment puis-je fermer mon Compte AWS ? .
  - Fermez un appareil autonome. Compte AWS

Tâches de facturation

- Activer l'accès IAM à la console de gestion de la facturation et des coûts.
- Certaines tâches de facturation sont limitées à l'utilisateur root. Consultez le guide de gestion Compte AWS d'un AWS Billing utilisateur intégré pour plus d'informations.
- Afficher certaines factures fiscales. Un utilisateur IAM disposant de l'ViewBillingautorisation aws-portal : peut consulter et télécharger les factures de TVA depuis AWS l'Europe, mais pas depuis AWS Inc. ou Amazon Internet Services Private Limited (AISPL).

AWS GovCloud (US) Tâches

- Inscrivez-vous à AWS GovCloud (US).
- Demandez les clés d'accès de l'utilisateur root au AWS GovCloud (US) compte auprès de AWS Support.

Tâche Amazon EC2

- Inscrit en tant que vendeur sur le Marketplace des instances réservées.

AWS KMS Tâche

- Si une AWS Key Management Service clé devient ingérable, un administrateur peut la récupérer en contactant AWS Support ; toutefois, il AWS Support répond au numéro de téléphone principal de votre utilisateur root pour obtenir l'autorisation en confirmant le ticket OTP.

Tâche Amazon Mechanical Turk

- Liez votre compte Compte AWS à votre compte mTurk Requester.

Tâches du service Amazon Simple Storage

- Configurer un compartiment Amazon S3 pour activer MFA (authentification multifactorielle).
- Modifiez ou supprimez une politique de compartiment Amazon S3 qui refuse tous les principes.

Tâche du service Amazon Simple Queue

- Modifiez ou supprimez une politique de ressources Amazon SQS qui refuse tous les principaux.

Résolution de problèmes relatifs à l'utilisateur root

Utilisez les informations fournies pour vous aider à résoudre les problèmes liés à l'utilisateur root d'un Compte AWS.

Je ne peux pas effectuer les tâches que je pense pouvoir effectuer lorsque je suis connecté en tant qu'utilisateur root du compte

Si vous ne parvenez pas à effectuer des tâches lorsque vous êtes connecté en tant qu'utilisateur root du compte, il se peut que votre compte soit membre d'une organisation dans AWS Organizations. Si c'est le cas, et que votre administrateur d'organisation utilise une politique de contrôle des services (SCP) pour limiter les autorisations de votre compte, tous les utilisateurs, y compris l'utilisateur root, sont concernés. Pour plus d'informations, veuillez consulter Politiques de contrôle de service du Guide de l'utilisateur AWS Organizations .

J'ai oublié le mot de passe utilisateur root de mon compte Compte AWS

Si vous êtes un utilisateur root et que vous avez perdu ou oublié le mot de passe de votre compte Compte AWS, vous pouvez le réinitialiser. Vous devez connaître l'adresse e-mail utilisée lors de la création de l' Compte AWS et vous devez avoir accès au compte de messagerie. Pour plus d'informations, consultez Réinitialisation d'un mot de passe d'utilisateur racine perdu ou oublié.

Je n'ai pas accès à l'e-mail de mon Compte AWS

Lorsque vous créez un Compte AWS, vous fournissez une adresse e-mail et un mot de passe. Ce sont les informations d'identification pour le Utilisateur racine d'un compte AWS. Si vous n'êtes pas sûr de l'adresse e-mail associée à votre Compte AWS, recherchez les messages envoyés depuis `@signin.aws` ou `@verify.signin.aws` vers une adresse e-mail de votre organisation qui aurait pu être utilisée pour ouvrir le Compte AWS.

Si vous connaissez l'adresse e-mail, mais que vous n'avez plus accès à l'e-mail, essayez d'abord de récupérer l'accès à l'e-mail en utilisant l'une des options suivantes :

- Si vous possédez le domaine pour l'adresse e-mail, vous pouvez restaurer une adresse e-mail supprimée. Vous pouvez également configurer un « catch-all » pour votre compte de messagerie,

qui « attrape tous » les messages envoyés à des adresses e-mail qui n'existent plus dans le serveur de messagerie et les redirige vers une autre adresse e-mail.

- Si l'adresse e-mail sur le compte fait partie de votre système de messagerie d'entreprise, nous vous recommandons de contacter vos administrateurs de système informatique. Ils peuvent vous aider à récupérer l'accès à l'e-mail.

Si vous ne parvenez toujours pas à vous connecter à votre compte Compte AWS, vous pouvez trouver d'autres options d'assistance sur Contactez-nous.

Informations connexes

Les articles suivants fournissent des informations supplémentaires sur l'utilisation de l'utilisateur root.

- Quelles sont les meilleures pratiques pour sécuriser mes ressources Compte AWS et les siennes ?
- Comment créer une règle d' EventBridge événement pour m'avertir que mon utilisateur root a été utilisé ?
- Surveiller et notifier l' Utilisateur racine d'un compte AWS activité
- Surveiller l'activité de l'utilisateur root IAM

Utilisateurs IAM

> **Important**
>
> Les meilleures pratiques IAM recommandent de demander aux utilisateurs humains d'utiliser la fédération avec un fournisseur d'identité pour accéder à l'aide d'informations d'identification temporaires plutôt que d' AWS utiliser des utilisateurs IAM dotés d'informations d'identification à long terme.

Un utilisateur AWS Identity and Access Management (IAM) est une entité que vous créez dans AWS. L'utilisateur IAM représente l'utilisateur humain ou la charge de travail qui utilise l'utilisateur IAM pour interagir avec lui. AWS Un utilisateur se AWS compose d'un nom et d'informations d'identification.

Un utilisateur IAM doté d'autorisations d'administrateur n'est pas un Utilisateur racine d'un compte AWS. Pour de plus amples informations sur l'utilisateur racine, veuillez consulter Utilisateur racine d'un compte AWS.

Comment AWS identifier un utilisateur IAM

Lorsque vous créez un utilisateur IAM, IAM implémente les éléments suivants pour l'identifier :

- Un « nom convivial » pour l'utilisateur IAM. Il s'agit du nom que vous avez spécifié lors de la création de l'utilisateur IAM, par exemple `Richard` ou `Anaya`. Ce sont les noms que vous voyez dans AWS Management Console.
- Un Amazon Resource Name (ARN) pour l'utilisateur IAM. Vous utilisez l'ARN lorsque vous devez identifier de manière unique l'utilisateur IAM dans l' AWS ensemble. Par exemple, vous pouvez utiliser un ARN pour spécifier l'utilisateur IAM en tant que `Principal` dans la politique IAM d'un compartiment Amazon S3. L'ARN d'un utilisateur IAM peut se présenter comme suit :

  `arn:aws:iam::`*`account-ID-without-hyphens`*`:user/Richard`
- Un identifiant unique pour l'utilisateur IAM. Cet ID est renvoyé uniquement lorsque vous utilisez l'API, Tools for Windows PowerShell ou AWS CLI pour créer l'utilisateur IAM ; il ne figure pas dans la console.

Pour plus d'informations sur ces identifiants, consultez Identifiants IAM.

Utilisateurs IAM et informations d'identification

Vous pouvez y accéder de différentes manières AWS en fonction des informations d'identification de l'utilisateur IAM :

- Mot de passe de la console : un mot de passe que l'utilisateur IAM peut entrer pour se connecter à des sessions interactives telles que l' AWS Management Console. La désactivation du mot de passe (accès à la console) pour un utilisateur IAM l'empêche de se connecter à l' AWS Management Console aide de ses informations d'identification. Cela ne modifie pas ses autorisations ni ne l'empêche d'accéder à la console à l'aide d'un rôle endossé.
- Clés d'accès : utilisées pour passer des appels programmatiques à AWS. Toutefois, il existe des alternatives plus sécurisées à envisager avant de créer des clés d'accès pour les utilisateurs IAM. Pour plus d'informations, consultez Considérations et alternatives relatives aux clés d'accès à long terme dans Références générales AWS. Si l'utilisateur IAM dispose de clés d'accès actives, celles-ci continuent de fonctionner et autorisent l'accès via les AWS CLI Outils pour Windows PowerShell, AWS l'API ou l'Application AWS Console Mobile.
- Clés SSH à utiliser avec CodeCommit : clé publique SSH au format OpenSSH qui peut être utilisée pour s'authentifier auprès de. CodeCommit

- Certificats de serveur : certificats SSL/TLS que vous pouvez utiliser pour vous authentifier auprès de certains services. AWS Nous vous recommandons d'utiliser AWS Certificate Manager (ACM) pour provisionner, gérer et déployer vos certificats de serveur. Utilisez IAM uniquement lorsque vous devez prendre en charge des connexions HTTPS dans une région non prise en charge par ACM. Pour savoir quelles régions prennent en charge ACM, consultez Points de terminaison et quotas AWS Certificate Manager dans Références générales AWS.

Vous pouvez choisir les informations d'identification qui conviennent à votre utilisateur IAM. Lorsque vous utilisez la AWS Management Console pour créer un utilisateur IAM, vous devez au moins choisir d'inclure un mot de passe ou des clés d'accès à la console. Par défaut, un tout nouvel utilisateur IAM créé à l'aide de l' AWS API AWS CLI or ne possède aucune information d'identification. Vous devez créer le type d'informations d'identification pour un utilisateur IAM en fonction de votre cas d'utilisation.

Vous disposez des options suivantes pour administrer les mots de passe, les clés d'accès et les dispositifs d'authentification multifactorielle (MFA) :

- Gérer les mots de passe pour vos utilisateurs IAM. Créez et modifiez les mots de passe permettant d'accéder à AWS Management Console. Définissez une politique de mot de passe afin d'appliquer une complexité minimale pour les mots de passe. Autorisez les utilisateurs à modifier leurs propres mots de passe.
- Gérer les clés d'accès pour vos utilisateurs IAM. Créez et mettez à jour les clés d'accès afin de permettre l'accès par programmation aux ressources de votre compte.
- Activez l'authentification multifactorielle (MFA) pour l'utilisateur IAM. Selon les bonnes pratiques, nous vous recommandons d'exiger une authentification multifactorielle pour tous les utilisateurs IAM de votre compte. Avec MFA, les utilisateurs doivent fournir deux formes d'identification : tout d'abord, ils fournissent les informations d'identification faisant partie de leur identité utilisateur (un mot de passe ou une clé d'accès). En outre, ils fournissent un code numérique temporaire généré sur un matériel ou par une application sur un smartphone ou une tablette.
- Recherche de mots de passe et clés d'accès inutilisés. Toute personne disposant d'un mot de passe ou de clés d'accès pour votre compte ou d'un utilisateur IAM de votre compte a accès à vos AWS ressources. En matière de sécurité, les bonnes pratiques consistent à supprimer les mots de passe et les clés d'accès dont les utilisateurs n'ont plus besoin.
- Téléchargez un rapport d'informations d'identification pour votre compte. Vous pouvez générer et télécharger un rapport sur les informations d'identification qui répertorie tous les utilisateurs IAM de votre compte et le statut de leurs diverses informations d'identification, notamment leurs mots

de passe, clés d'accès et dispositifs MFA. Pour les mots de passe et les clés d'accès, le rapport d'informations d'identification indique leur dernière date d'utilisation.

Rôles et autorisations IAM

Par défaut, un nouvel utilisateur IAM ne dispose d'aucune autorisation pour faire quoi que ce soit. Ils ne sont pas autorisés à effectuer des AWS opérations ou à accéder à des AWS ressources. Lorsque vous configurez des utilisateurs IAM individuels, il est également possible de leur affecter des autorisations individuellement. Vous pouvez attribuer des autorisations administratives à quelques utilisateurs, qui pourront ensuite administrer vos AWS ressources et même créer et gérer d'autres utilisateurs IAM. Dans la plupart des cas, toutefois, vous souhaitez limiter les autorisations d'un utilisateur aux seules tâches (AWS actions ou opérations) et aux ressources nécessaires à la tâche.

Prenons l'exemple d'un utilisateur nommé Diego. Lorsque vous créez l'utilisateur IAM `Diego`, vous lui créez un mot de passe et lui attachez des autorisations qui lui permettent de lancer une instance Amazon EC2 spécifique et de lire (`GET`) les informations d'une table dans une base de données Amazon RDS. Pour les procédures relatives à la création des utilisateurs et à l'octroi d'informations d'identification initiales, consultez Création d'un utilisateur IAM dans votre Compte AWS. Pour les procédures relatives à la modification des autorisations pour les utilisateurs existants, consultez Modification des autorisations pour un utilisateur IAM. Pour les procédures relatives à la modification des clés d'accès et du mot de passe de l'utilisateur, consultez Gestion des mots de passe utilisateur dans AWS et Gestion des clés d'accès pour les utilisateurs IAM.

Vous pouvez également ajouter une limite des autorisations à vos utilisateurs IAM. Une limite d'autorisations est une fonctionnalité avancée qui vous permet d'utiliser des politiques AWS gérées pour limiter le maximum d'autorisations qu'une politique basée sur l'identité peut accorder à un utilisateur ou à un rôle IAM. Pour plus d'informations sur les types de politiques et les utilisations, veuillez consulter Politiques et autorisations dans IAM.

Utilisateurs et comptes IAM

Chaque utilisateur IAM est associé à un seul et même Compte AWS. Les utilisateurs IAM étant définis au sein de votre entreprise Compte AWS, ils n'ont pas besoin de disposer d'un mode de paiement enregistré auprès AWS de. Toute AWS activité effectuée par les utilisateurs IAM sur votre compte est facturée sur votre compte.

Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d’informations, consultez IAM et quotas AWS STS.

Utilisateurs IAM en tant que comptes de service

Un utilisateur IAM est une ressource dans IAM à laquelle des informations d'identification et ses autorisations sont associées. Un utilisateur IAM peut représenter une personne ou une application qui utilise ses informations d'identification pour exécuter des demandes AWS . En général, ceci est appelé un compte de service. Si vous choisissez d'utiliser les informations d'identification à long terme d'un utilisateur IAM dans votre application, n'incorporez pas directement les clés d'accès dans le code de votre application. Les AWS SDK et le vous AWS Command Line Interface permettent de placer les clés d'accès dans des emplacements connus afin de ne pas avoir à les conserver dans le code. Pour de plus amples informations, consultez Gestion correcte des clés d'accès utilisateur IAM (français non garanti) dans Références générales AWS. En tant que bonne pratique, vous pouvez également utiliser des informations d'identification de sécurité temporaires (rôles IAM) au lieu des clés d'accès à long terme.

Création d'un utilisateur IAM dans votre Compte AWS

Follow us on Twitter

Important

Les meilleures pratiques IAM recommandent de demander aux utilisateurs humains d'utiliser la fédération avec un fournisseur d'identité pour accéder à l'aide d'informations d'identification temporaires plutôt que d' AWS utiliser des utilisateurs IAM dotés d'informations d'identification à long terme.

Note

Si vous avez trouvé cette page parce que vous recherchez des informations sur le Product Advertising API pour vendre des produits Amazon sur votre site web, veuillez consulter la documentation du Product Advertising API version 5.0.
Si vous avez accédé à cette page via la console IAM, il est possible que votre compte n'inclut pas d'utilisateurs IAM, bien que vous soyez connecté. Vous pouvez vous être connecté en tant qu' Utilisateur racine d'un compte AWSà l'aide d'un rôle ou avec des informations d'identification temporaires. Pour en savoir plus sur ces identités IAM, consultez Identités IAM (utilisateurs, groupes d'utilisateurs et rôles).

Le processus de création d'un utilisateur et sa capacité à exécuter des tâches se compose des étapes suivantes :

1. Créez l'utilisateur dans AWS Management Console les AWS CLI outils pour Windows PowerShell ou à l'aide d'une opération d' AWS API. Si vous créez l'utilisateur dans le AWS Management Console, les étapes 1 à 4 sont gérées automatiquement, en fonction de vos choix. Si vous créez les utilisateurs par programme, vous devez exécuter chacune de ces étapes individuellement.
2. Créez les informations d'identification pour l'utilisateur, en fonction du type d'accès dont il a besoin :
   - Activer l'accès à la console — facultatif : si l'utilisateur doit accéder au AWS Management Console, créez un mot de passe pour l'utilisateur. La désactivation de l'accès à la console pour un utilisateur l'empêche de se connecter à l' AWS Management Console à l'aide de son nom d'utilisateur et de son mot de passe. Cela ne modifie pas ses autorisations ni ne l'empêche d'accéder à la console à l'aide d'un rôle endossé.

   > **Tip**
   >
   > Créez uniquement les informations d'identification dont a besoin l'utilisateur. Par exemple, pour un utilisateur qui a uniquement besoin d'un accès via le AWS Management Console, ne créez pas de clés d'accès.

3. Donnez à l'utilisateur les autorisations concernant les tâches requises en l'ajoutant à un ou plusieurs groupes. Vous pouvez également accorder des autorisations en attachant des politiques d'autorisations directement à l'utilisateur. Toutefois, nous vous recommandons plutôt de placer vos utilisateurs dans des groupes et de gérer leurs autorisations par le biais de politiques attachées à ces groupes. Vous pouvez également utiliser une limite d'autorisations pour restreindre les autorisations dont peut disposer un utilisateur, même si cela n'est pas courant.
4. (Facultatif) Ajoutez des métadonnées à l'utilisateur en associant des balises. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Balisage des ressources IAM.
5. Fournissez à l'utilisateur les informations nécessaires à la connexion. Cela inclut le mot de passe et l'URL de la console de la page de connexion au compte sur laquelle l'utilisateur saisit ces informations d'identification. Pour plus d’informations, veuillez consulter Comment les utilisateurs d'IAM se connectent à AWS.

6. (Facultatif) Configurez l'authentification multifacteur (MFA) pour l'utilisateur. La MFA demande à l'utilisateur de fournir un one-time-use code chaque fois qu'il se connecte au. AWS Management Console
7. (Facultatif) Accordez aux utilisateurs les autorisations requises pour gérer leurs propres informations d'identification. (Par défaut, les utilisateurs ne sont pas autorisés à gérer leurs propres informations d'identification.) Pour plus d'informations, veuillez consulter Autorisation des utilisateurs IAM à modifier leurs propres mots de passe.

Pour plus d'informations sur les autorisations requises pour créer un utilisateur, consultez Autorisations requises pour accéder aux autres ressources IAM.

Rubriques

- Création d'utilisateurs IAM (console)
- Création d'utilisateurs IAM (AWS CLI)
- Création d'utilisateurs IAM (AWS API)

Création d'utilisateurs IAM (console)

Vous pouvez utiliser le AWS Management Console pour créer des utilisateurs IAM.

Pour créer un utilisateur IAM (console)

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS du Guide de l'utilisateur Connexion àAWS .
2. Sur la page d'accueil de la console, sélectionnez le service IAM.
3. Dans le volet de navigation, sélectionnez Utilisateurs, puis Ajouter des utilisateurs.
4. Sur la page Spécifier les détails de l'utilisateur, sous Détails de l'utilisateur, dans Nom d'utilisateur, entrez le nom du nouvel utilisateur. Il s'agit de son nom de connexion pour AWS.

> **Note**
>
> Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d'informations, consultez IAM et quotas AWS STS. Les noms d'utilisateur peuvent combiner jusqu'à 64 lettres, chiffres et caractères suivants : plus (+), égal (=), virgule (,), point (.), arobase (@), trait de soulignement (_) et tiret (-). Les noms doivent être uniques dans un compte. Ils ne sont pas sensibles à la casse. Par exemple, vous ne pouvez pas

créer deux utilisateurs nommés TESTUSER et testuser. Lorsqu'un nom d'utilisateur est utilisé dans une politique ou dans le cadre d'un ARN, il est sensible à la casse. Lorsqu'un nom d'utilisateur apparaît aux clients dans la console, par exemple lors du processus de connexion, il n'est pas sensible à la casse.

5. Sélectionnez Fournir un accès utilisateur au — AWS Management Console facultatif Cela produit des AWS Management Console informations d'identification pour le nouvel utilisateur.

   Il vous est demandé si vous accordez l'accès à la console à un utilisateur. Nous vous recommandons de créer des utilisateurs dans IAM Identity Center plutôt que dans IAM.

   - Pour créer l'utilisateur dans IAM Identity Center, sélectionnez Spécifier un utilisateur dans Identity Center.

     Si vous n'avez pas activé IAM Identity Center, la sélection de cette option vous permet d'accéder à la page de service de la console pour le faire. Pour plus de détails sur cette procédure, voir Commencer à exécuter les tâches courantes dans IAM Identity Center dans le guide de l'AWS IAM Identity Center utilisateur

     Si vous avez activé IAM Identity Center, sélectionnez cette option pour accéder à la page Spécifier un utilisateur dans IAM Identity Center. Pour plus de détails sur cette procédure, voir Ajouter des utilisateurs dans le guide de AWS IAM Identity Center l'utilisateur
   - Si vous ne pouvez pas utiliser IAM Identity Center, sélectionnez Je souhaite créer un utilisateur IAM et poursuivez cette procédure.

   a. Pour Mot de passe de la console, sélectionnez l'une des options suivantes :

      - Mot de passe généré automatiquement – L'utilisateur obtient un mot de passe généré de façon aléatoire qui correspond à la politique de mot de passe de compte. Vous pouvez afficher ou télécharger le mot de passe lorsque vous accédez à la page Récupérer le mot de passe.
      - Mot de passe personnalisé – L'utilisateur se voit attribuer le mot de passe que vous entrez dans la zone.

   b. (Facultatif) L'option Les utilisateurs doivent créer un nouveau mot de passe à leur prochaine connexion (recommandée) est sélectionnée par défaut afin d'obliger l'utilisateur à changer son mot de passe lors de sa première connexion.

> **Note**
>
> Si un administrateur a activé le paramètre de politique de mot de passe de compte (Autoriser les utilisateurs à modifier leur propre mot de passe), cette case à cocher ne sert à rien. Dans le cas contraire, il attache automatiquement une politique AWS gérée nommée `IAMUserChangePassword` aux nouveaux utilisateurs. La politique leur accorde l'autorisation de modifier leurs propres mots de passe.

6. Sélectionnez Suivant.
7. Sur la page Définir les autorisations, spécifiez la façon dont vous souhaitez attribuer des autorisations à cet utilisateur. Sélectionnez l'une des trois options suivantes :

   - Ajouter un utilisateur au groupe – Sélectionnez cette option si vous souhaitez attribuer l'utilisateur à un ou plusieurs groupes disposant déjà des politiques d'autorisations. IAM affiche une liste des groupes de votre compte ainsi que les politiques attachées. Vous pouvez sélectionner un ou plusieurs groupes existants, ou sélectionner Créer un groupe pour créer un groupe. Pour plus d'informations, consultez Modification des autorisations pour un utilisateur IAM.
   - Copier les autorisations – Sélectionnez cette option pour copier toutes les appartenances au groupe, les politiques gérées attachées et les politiques en ligne intégrées et toutes les limites d'autorisations existantes d'un utilisateur existant vers de nouveaux utilisateurs. IAM affiche une liste des utilisateurs de votre compte. Sélectionnez celui dont les autorisations correspondent le plus aux besoins de votre nouvel utilisateur.
   - Joindre directement les politiques : sélectionnez cette option pour afficher la liste des politiques AWS gérées et gérées par le client dans votre compte. Sélectionnez les politiques que vous souhaitez attacher à l'utilisateur ou sélectionnez Créer une politique pour ouvrir un nouvel onglet de navigateur et créer une toute nouvelle politique. Pour plus d'informations, consultez l'étape 4 de la procédure Création de politiques IAM. Après avoir créé la politique, fermez cet onglet et revenez à votre onglet d'origine pour ajouter la politique à l'utilisateur.

   > **Tip**
   >
   > Chaque fois que cela est possible, affectez vos stratégies à un groupe, puis faites de ces utilisateurs des membres des groupes appropriés.

8. (Facultatif) Définissez une limite d'autorisations. Il s'agit d'une fonctionnalité avancée.

Ouvrez la section Définir une limite d'autorisations et sélectionnez Utiliser une limite d'autorisations pour contrôler le nombre maximum d'autorisations. IAM affiche une liste des politiques AWS gérées et gérées par le client dans votre compte. Sélectionnez la politique à utiliser pour la limite d'autorisations ou sélectionnez Créer une politique pour ouvrir un nouvel onglet de navigateur et créer une nouvelle politique. Pour plus d'informations, consultez l'étape 4 de la procédure Création de politiques IAM. Une fois la politique créée, fermez cet onglet et revenez à l'onglet initial pour sélectionner la politique à utiliser pour la limite d'autorisations.

9. Sélectionnez Suivant.
10. (Facultatif) Sur la page Vérifier et créer, sous Balises, sélectionnez Ajouter une nouvelle balise pour ajouter des métadonnées à l'utilisateur en associant les balises sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Balisage des ressources IAM.
11. Vérifiez tous les choix que vous avez faits jusqu'à présent. Une fois que vous êtes prêt à continuer, sélectionnez Créer un utilisateur.
12. Sur la page Récupérer le mot de passe, récupérez le mot de passe attribué à l'utilisateur :

    - Sélectionnez Afficher à côté du mot de passe pour afficher le mot de passe de l'utilisateur et l'enregistrer manuellement.
    - Sélectionnez Télécharger au format .csv pour télécharger les informations de connexion de l'utilisateur sous forme fichier .csv que vous pouvez enregistrer dans un emplacement sûr.
13. Sélectionnez Instructions de connexion par e-mail. Votre client de messagerie local s'ouvre avec un modèle que vous pouvez personnaliser et envoyer à l'utilisateur. Le modèle d'e-mail inclut les informations suivantes pour chaque utilisateur :

    - Nom utilisateur
    - URL de la page de connexion au compte. Utilisez l'exemple suivant, en remplaçant l'ID et l'alias de compte comme approprié :

    ```
    https://AWS-account-ID or alias.signin.aws.amazon.com/console
    ```

> **Important**
>
> Le mot de passe de l'utilisateur n'est pas inclus dans l'e-mail généré. Vous devez fournir le mot de passe à l'utilisateur d'une manière conforme aux normes de sécurité de votre organisation.

14. Si l'utilisateur a également besoin de clés d'accès pour accéder par programmation, reportez-vous àGestion des clés d'accès pour les utilisateurs IAM.

Création d'utilisateurs IAM (AWS CLI)

Vous pouvez utiliser le AWS CLI pour créer un utilisateur IAM.

Pour créer un utilisateur IAM (AWS CLI)

1. Créez un utilisateur.

   - aws iam create-user

2. (Facultatif) Donnez à utilisateur l'accès à l'interface AWS Management Console. Un mot de passe est requis. Vous devez également fournir à l'utilisateur l'URL de la page de connexion à votre compte.

   - était un objectif create-login-profile

3. (Facultatif) Donnez à l'utilisateur à un accès par programme. Cela nécessite des clés d'accès.

   - était un objectif create-access-key
   - Outils pour Windows PowerShell : New-IAM AccessKey
   - API IAM : CreateAccessKey

> **Important**
>
> Il s'agit de votre seule opportunité de consulter ou de télécharger les clés d'accès secrètes, et vous devez fournir ces informations à vos utilisateurs avant qu'ils puissent utiliser l' AWS API. Enregistrez les nouveaux ID de clé d'accès et clé d'accès secrète de l'utilisateur dans un endroit sûr et sécurisé. Vous ne pourrez plus accéder aux clés d'accès secrètes après cette étape.

4. Ajoutez l'utilisateur à un ou plusieurs groupes. Les stratégies attachées aux groupes que vous spécifiez doivent accorder les autorisations appropriées à l'utilisateur.

   - était un objectif add-user-to-group

5. (facultatif) Attachez une politique à l'utilisateur afin de définir ses autorisations. Remarque : Nous vous recommandons de gérer les autorisations d'un utilisateur en l'ajoutant à un groupe, puis en attachant une politique au groupe plutôt que directement au niveau de l'utilisateur.

   - était un objectif attach-user-policy

6. (Facultatif) Ajoutez des attributs personnalisés à l'utilisateur en associant des balises. Pour plus d'informations, veuillez consulter Gestion des balises sur les utilisateurs IAM (AWS CLI ou AWS API).

7. (Facultatif) Accordez à l'utilisateur l'autorisation requise pour gérer ses propres informations d'identification. Pour plus d'informations, consultez AWS: permet aux utilisateurs IAM authentifiés par MFA de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

Création d'utilisateurs IAM (AWS API)

Vous pouvez utiliser l' AWS API pour créer un utilisateur IAM.

Pour créer un utilisateur IAM à partir de (AWS API)

1. Créez un utilisateur.

   - CreateUser

2. (Facultatif) Donnez à utilisateur l'accès à l'interface AWS Management Console. Un mot de passe est requis. Vous devez également fournir à l'utilisateur l'URL de la page de connexion à votre compte.

   - CreateLoginProfile

3. (Facultatif) Donnez à l'utilisateur à un accès par programme. Cela nécessite des clés d'accès.

   - CreateAccessKey

> **Important**
>
> Il s'agit de votre seule opportunité de consulter ou de télécharger les clés d'accès secrètes, et vous devez fournir ces informations à vos utilisateurs avant qu'ils puissent utiliser l' AWS API. Enregistrez les nouveaux ID de clé d'accès et clé d'accès secrète de l'utilisateur dans un endroit sûr et sécurisé. Vous ne pourrez plus accéder aux clés d'accès secrètes après cette étape.

4. Ajoutez l'utilisateur à un ou plusieurs groupes. Les stratégies attachées aux groupes que vous spécifiez doivent accorder les autorisations appropriées à l'utilisateur.

   - AddUserToGroup

5. (facultatif) Attachez une politique à l'utilisateur afin de définir ses autorisations. Remarque : Nous vous recommandons de gérer les autorisations d'un utilisateur en l'ajoutant à un groupe, puis en attachant une politique au groupe plutôt que directement au niveau de l'utilisateur.

   - AttachUserPolicy

6. (Facultatif) Ajoutez des attributs personnalisés à l'utilisateur en associant des balises. Pour plus d'informations, veuillez consulter Gestion des balises sur les utilisateurs IAM (AWS CLI ou AWS API).
7. (Facultatif) Accordez à l'utilisateur l'autorisation requise pour gérer ses propres informations d'identification. Pour plus d'informations, voir AWS: permet aux utilisateurs IAM authentifiés par MFA de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

Contrôler l'accès des utilisateurs IAM à l' AWS Management Console

Les utilisateurs IAM autorisés qui se connectent à vous Compte AWS via le AWS Management Console peuvent accéder à vos AWS ressources. La liste suivante indique comment vous pouvez autoriser les utilisateurs IAM à accéder à vos Compte AWS ressources via le AWS Management Console. Il montre également comment les utilisateurs d'IAM peuvent accéder aux autres fonctionnalités du AWS compte via le AWS site Web.

> **Note**
>
> L'utilisation d'IAM n'induit aucuns frais.

Le AWS Management Console

Vous créez un mot de passe pour chaque utilisateur IAM devant accéder à l' AWS Management Console. Les utilisateurs accèdent à la console via votre page de Compte AWS connexion compatible IAM. Pour plus d'informations sur l'accès à la page de connexion, consultez Connexion à AWS dans le Guide de l'utilisateur Connexion à AWS . Pour plus d'informations sur la création de mots de passe, consultez Gestion des mots de passe utilisateur dans AWS.

Vous pouvez empêcher un utilisateur IAM d'accéder au AWS Management Console en supprimant son mot de passe. Cela les empêche de se connecter à l' AWS Management Console aide de leurs identifiants de connexion. Cela ne modifie pas ses autorisations ni ne l'empêche d'accéder à la console à l'aide d'un rôle endossé. Si l'utilisateur dispose de clés d'accès actives, elles continuent de fonctionner et autorisent l' AWS CLI accès via les Outils pour Windows PowerShell, l' AWS API ou l'Application AWS Console Mobile.

Vos AWS ressources, telles que les instances Amazon EC2, les compartiments Amazon S3, etc.

Même si vos utilisateurs IAM ont des mots de passe, ils doivent également disposer d'une autorisation leur permettant d'accéder à vos ressources AWS . Lorsque vous créez un utilisateur IAM, celui-ci ne dispose par défaut d'aucune autorisation. Pour accorder à vos utilisateurs IAM les autorisations dont ils ont besoin, vous leur attachez des politiques. Si plusieurs utilisateurs IAM effectuent les mêmes tâches avec les mêmes ressources, vous pouvez les affecter à un groupe. Affectez ensuite les autorisations à ce groupe. Pour plus d'informations sur la création d'utilisateurs et de groupes IAM, veuillez consulter la rubrique Identités IAM (utilisateurs, groupes d'utilisateurs et rôles). Pour plus d'informations sur l'utilisation de stratégies pour l'octroi d'autorisations, consultez Gestion de l'accès aux AWS ressources.

AWS Forums de discussion

Tout le monde peut lire les messages publiés sur les Forums de discussion AWS. Les utilisateurs qui souhaitent publier des questions ou des commentaires AWS sur le forum de discussion peuvent le faire en utilisant leur nom d'utilisateur. La première fois qu'un utilisateur publie AWS sur le forum de discussion, il est invité à saisir un surnom et une adresse e-mail. Seul cet utilisateur peut utiliser ce surnom dans les forums de AWS discussion.

Vos informations Compte AWS de facturation et d'utilisation

Vous pouvez autoriser les utilisateurs à accéder à vos informations Compte AWS de facturation et d'utilisation. Pour de plus amples informations, veuillez consulter Contrôle de l'accès à vos informations de facturation dans le Guide de l'utilisateur AWS Billing .

Informations Compte AWS de votre profil

Les utilisateurs ne peuvent pas accéder aux informations Compte AWS de votre profil.

Vos informations Compte AWS de sécurité

Les utilisateurs ne peuvent pas accéder à vos informations Compte AWS de sécurité.

> **Note**
>
> Les politiques IAM contrôlent l'accès, quelle que soit l'interface. Par exemple, vous pouvez fournir à un utilisateur un mot de passe pour accéder au AWS Management Console. Les politiques de cet utilisateur (ou de tout groupe auquel appartient l'utilisateur) contrôleraient ce que l'utilisateur peut faire dans le AWS Management Console. Vous pouvez également fournir à l'utilisateur des clés d' AWS accès pour effectuer des appels d'API à AWS. Les politiques contrôleraient les actions que l'utilisateur pourrait appeler via une bibliothèque ou un client qui utilise ces clés d'accès pour l'authentification.

Comment les utilisateurs d'IAM se connectent à AWS

Pour vous connecter en AWS Management Console tant qu'utilisateur IAM, vous devez fournir votre identifiant de compte ou votre alias de compte en plus de votre nom d'utilisateur et de votre mot de passe. Lorsque votre administrateur a créé votre utilisateur IAM dans la console, il a dû vous envoyer vos informations d'identification de connexion, notamment votre nom d'utilisateur et l'URL de la page de connexion à votre compte, qui inclut votre ID de compte ou votre alias de compte.

```
https://My_AWS_Account_ID.signin.aws.amazon.com/console/
```

> **Conseil**
>
> Pour créer un marque-page pour la page de connexion à votre compte dans votre navigateur web, vous devez saisir manuellement l'URL de connexion de votre compte lors de la création du marque-page. N'utilisez pas la fonction « Marquer cette page » de votre navigateur web, en raison des redirections qui risquent de masquer l'URL de connexion.

Vous pouvez également vous connecter au point de terminaison général suivant et saisir manuellement votre ID de compte ou votre alias de compte :

```
https://console.aws.amazon.com/
```

Pour plus de commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser le nom d'utilisateur IAM et les informations de compte. La prochaine fois que l'utilisateur accède à une page du AWS Management Console, la console utilise le cookie pour le rediriger vers la page de connexion au compte.

Vous n'avez accès qu'aux AWS ressources spécifiées par votre administrateur dans la politique associée à votre identité d'utilisateur IAM. Pour travailler dans la console, vous devez disposer des autorisations nécessaires pour effectuer les actions effectuées par la console, telles que la liste et la création de AWS ressources. Pour plus d'informations, consultez Gestion de l'accès aux AWS ressources et Exemples de politiques basées sur l'identité IAM.

> **Note**
>
> Si votre organisation dispose déjà d'un système d'identité, vous souhaiterez peut-être créer une option d'authentification unique (SSO). Le SSO permet aux utilisateurs d'accéder AWS Management Console à votre compte sans qu'ils aient besoin d'une identité d'utilisateur IAM. L'authentification unique élimine également la nécessité pour les utilisateurs de se connecter au site de votre organisation et de s'y connecter AWS séparément. Pour plus d'informations, consultez Activation de l'accès à la AWS console par un courtier d'identité personnalisé.

Enregistrement des informations de connexion dans CloudTrail

Si vous activez CloudTrail l'enregistrement des événements de connexion dans vos journaux, vous devez savoir comment CloudTrail choisit où enregistrer les événements.

- Si les utilisateurs se connectent directement à une console, ils sont redirigés vers un point de terminaison de connexion international ou régional, si la console de service sélectionnée prend en charge les régions. Par exemple, la page d'accueil de la console principale prend en charge les régions, donc si vous vous connectez à l'URL suivante :

  ```
  https://alias.signin.aws.amazon.com/console
  ```

  vous êtes redirigé vers un point de connexion régional tel que`https://us-east-2.signin.aws.amazon.com`, ce qui entraîne une entrée de CloudTrail journal régional dans le journal régional de l'utilisateur :

En revanche, la console Amazon S3 ne prend pas en charge les régions, donc si vous vous connectez à l'URL suivante

```
https://alias.signin.aws.amazon.com/console/s3
```

AWS vous redirige vers le point de terminaison de connexion global à l'adresse`https://signin.aws.amazon.com`, ce qui entraîne une entrée de CloudTrail journal globale.

- Vous pouvez demander manuellement un point de terminaison de connexion régional spécifique en vous connectant à la page d'accueil régionale de la console principale à l'aide d'une syntaxe d'URL similaire à ce qui suit :

```
https://alias.signin.aws.amazon.com/console?region=ap-southeast-1
```

AWS vous redirige vers le point de connexion `ap-southeast-1` régional et entraîne un événement de CloudTrail journal régional.

Pour plus d'informations sur IAM CloudTrail et IAM, consultez la section Journalisation des événements IAM avec. CloudTrail

Si les utilisateurs ont besoin d'un accès par programmation pour utiliser votre compte, vous pouvez créer une paire de clés d'accès (un ID de clé d'accès et une clé d'accès secrète) pour chaque utilisateur. Toutefois, il existe des alternatives plus sécurisées à envisager avant de créer des clés d'accès pour les utilisateurs. Pour plus d'informations, consultez Considérations et alternatives relatives aux clés d'accès à long terme dans Références générales AWS.

Utilisation de dispositifs MFA avec votre page de connexion IAM

Les utilisateurs pour lesquels l'authentification multifactorielle (MFA) est configurée doivent utiliser leurs dispositifs MFA pour se connecter à l' AWS Management Console. Une fois que l'utilisateur a saisi ses informations de connexion, AWS vérifie dans son compte si le MFA est requis pour cet utilisateur. Les rubriques suivantes fournissent des informations sur la façon dont les utilisateurs effectuent leur connexion lorsque l'authentification MFA est requise.

Rubriques

- Connexion avec plusieurs dispositifs MFA activés
- Connexion avec une clé de sécurité FIDO

- Connexion avec un dispositif MFA virtuel
- Connexion avec un jeton TOTP matériel

Connexion avec plusieurs dispositifs MFA activés

Si un utilisateur se connecte en AWS Management Console tant qu'utilisateur Compte AWS root ou utilisateur IAM avec plusieurs appareils MFA activés pour ce compte, il n'a besoin d'utiliser qu'un seul appareil MFA pour se connecter. Une fois que l'utilisateur s'est authentifié à l'aide du mot de passe de l'utilisateur, il sélectionne le type de dispositif MFA qu'il souhaite utiliser pour terminer l'authentification. L'utilisateur est ensuite invité à s'authentifier avec le type de dispositif qu'il a sélectionné.

Connexion avec une clé de sécurité FIDO

Si l'authentification MFA est requise pour l'utilisateur, une deuxième page de connexion s'affiche. L'utilisateur a besoin d'utiliser la clé de sécurité FIDO.

> **Note**
>
> Les utilisateurs de Google Chrome ne doivent choisir aucune des options disponibles dans la fenêtre contextuelle qui demande Verify your identity with amazon.com (Vérifiez votre identité auprès d'amazon.com). Il vous suffit d'appuyer sur la clé de sécurité.

Contrairement aux autres dispositifs MFA, les clés de sécurité FIDO ne se désynchronisent pas. Les administrateurs peuvent désactiver une clé de sécurité FIDO si elle est perdue ou endommagée. Pour plus d’informations, consultez Désactivation des dispositifs MFA (console).

Pour plus d'informations sur les navigateurs compatibles WebAuthn et les appareils compatibles FIDO compatibles, AWS consultez. Configurations prises en charge pour l'utilisation des clés d'accès et des clés de sécurité

Connexion avec un dispositif MFA virtuel

Si l'authentification MFA est requise pour l'utilisateur, une deuxième page de connexion s'affiche. Dans le champ Code MFA, l'utilisateur doit entrer le code numérique fourni par l'application MFA.

Si le code MFA est correct, l'utilisateur peut accéder à l’interface AWS Management Console. Si le code est incorrect, l'utilisateur peut refaire une tentative avec un autre code.

Un dispositif MFA virtuel peut être désynchronisé. Si un utilisateur ne parvient pas à se connecter AWS Management Console après plusieurs tentatives, il est invité à synchroniser le périphérique MFA virtuel. L'utilisateur peut suivre les instructions affichées à l'écran pour synchroniser le dispositif MFA virtuel. Pour plus d'informations sur la façon dont vous pouvez synchroniser un appareil pour le compte d'un utilisateur de votre Compte AWS, consultezRésynchronisation de dispositifs MFA virtuels et matériels.

Connexion avec un jeton TOTP matériel

Si l'authentification MFA est requise pour l'utilisateur, une deuxième page de connexion s'affiche. Dans le champ Code MFA, l'utilisateur doit entrer le code numérique fourni par un jeton TOTP matériel.

Si le code MFA est correct, l'utilisateur peut accéder à l’interface AWS Management Console. Si le code est incorrect, l'utilisateur peut refaire une tentative avec un autre code.

Un jeton TOTP matériel peut être désynchronisé. Si un utilisateur ne parvient pas à se connecter AWS Management Console après plusieurs tentatives, il est invité à synchroniser le dispositif à jetons MFA. L'utilisateur peut suivre les instructions affichées à l'écran pour synchroniser le dispositif de jeton MFA. Pour plus d'informations sur la façon dont vous pouvez synchroniser un appareil pour le compte d'un utilisateur de votre Compte AWS, consultezRésynchronisation de dispositifs MFA virtuels et matériels.

Gestion des utilisateurs IAM

> **Note**
>
> En tant que bonne pratique, nous vous recommandons de demander aux utilisateurs humains d'utiliser la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires. Si vous suivez les bonnes pratiques, vous ne gérez pas les utilisateurs et les groupes IAM. Au lieu de cela, vos utilisateurs et vos groupes sont gérés en dehors de l'extérieur AWS et peuvent accéder aux AWS ressources en tant qu'identité fédérée. Une identité fédérée est un utilisateur de l'annuaire des utilisateurs de votre entreprise, un fournisseur d'identité Web, le AWS Directory Service, le répertoire Identity Center ou tout utilisateur qui accède AWS aux services à l'aide des informations d'identification fournies par le biais d'une source d'identité. Les identités fédérées utilisent les groupes définis par leur fournisseur d'identité. Si vous en utilisez AWS IAM Identity Center, consultez la section Gérer les identités dans IAM Identity Center dans le Guide de l'AWS IAM

Identity Center utilisateur pour plus d'informations sur la création d'utilisateurs et de groupes dans IAM Identity Center.

Amazon Web Services fournit plusieurs outils permettant de gérer les utilisateurs IAM dans votre Compte AWS. Vous pouvez répertorier les utilisateurs IAM de votre compte ou d'un groupe d'utilisateurs, ou dresser la liste de tous les groupes d'utilisateurs auxquels appartient un utilisateur. Vous pouvez renommer ou modifier le chemin d'accès d'un utilisateur IAM. Si vous passez à l'utilisation d'identités fédérées au lieu d'utilisateurs IAM, vous pouvez supprimer un utilisateur IAM de votre compte AWS ou désactiver l'utilisateur.

Pour plus d'informations sur l'ajout, la modification ou la suppression de politiques gérées pour un utilisateur IAM, consultez Modification des autorisations pour un utilisateur IAM. Pour plus d'informations sur la gestion des politiques en ligne pour les utilisateurs IAM, consultez Ajout et suppression d'autorisations basées sur l'identité IAM, Modification de politiques IAM et Suppression de politiques IAM. En guise de bonne pratique, utilisez des politiques gérées plutôt que des politiques en ligne. Les politiques gérées par AWS octroient des autorisations pour de nombreux cas d'utilisation courants. N'oubliez pas que les politiques AWS gérées peuvent ne pas accorder d'autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles peuvent être utilisées par tous les AWS clients. En conséquence, nous vous recommandons de réduire encore les autorisations en définissant des Politiques gérées par le client qui sont spécifiques à vos cas d'utilisation. Pour plus d’informations, consultez AWS politiques gérées. Pour plus d'informations sur les politiques AWS gérées conçues pour des fonctions professionnelles spécifiques, consultezAWS politiques gérées pour les fonctions professionnelles.

Pour en savoir plus sur la validation des politiques IAM, consultez Validation de politiques IAM.

Tip

IAM Access Analyzer peut analyser les services et les actions que vos rôles IAM utilisent, puis générer une politique précise que vous pouvez utiliser. Après avoir testé chaque stratégie générée, vous pouvez la déployer dans votre environnement de production. Cela garantit que vous n'accordez que les autorisations requises à vos charges de travail. Pour plus d'informations sur la génération de politiques, consultez IAM Access Analyzer policy generation.

Pour de plus amples informations sur la gestion des mots de passe utilisateur IAM, consultez Gestion des mots de passe des utilisateurs IAM.

Rubriques

- Afficher l'accès des utilisateurs
- Liste des utilisateurs IAM
- Renommer un utilisateur IAM
- Suppression d'un utilisateur IAM
- Désactivation d'un utilisateur IAM

Afficher l'accès des utilisateurs

Avant de supprimer un utilisateur, vous devez passer en revue son activité récente au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Liste des utilisateurs IAM

Vous pouvez répertorier les utilisateurs IAM de votre groupe d'utilisateurs IAM Compte AWS ou d'un groupe d'utilisateurs IAM spécifique, et répertorier tous les groupes d'utilisateurs auxquels appartient un utilisateur. Pour plus d'informations sur les autorisations requises pour répertorier des utilisateurs, consultez Autorisations requises pour accéder aux autres ressources IAM.

Pour répertorier tous les utilisateurs du compte

- AWS Management Console : dans le panneau de navigation, choisissez utilisateurs. La console affiche les utilisateurs de votre Compte AWS.
- AWS CLI : aws iam list-users
- AWS API : ListUsers

Pour répertorier les utilisateurs d'un groupe d'utilisateurs spécifique

- AWS Management Console : dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), le nom du groupe d'utilisateurs, puis l'onglet Users (Utilisateurs).
- AWS CLI : aws iam get-group

- AWS API : GetGroup

Pour répertorier tous les groupes d'utilisateurs auxquels appartient un utilisateur

- AWS Management Console : dans le panneau de navigation, sélectionnez Utilisateurs, choisissez le nom d'utilisateur, puis sélectionnez l'onglet Groupes.
- AWS CLI: cime list-groups-for-user
- AWS API : ListGroupsForUser

Renommer un utilisateur IAM

Pour modifier le nom ou le chemin d'un utilisateur, vous devez utiliser les AWS CLI outils pour Windows PowerShell ou AWS l'API. La console ne comporte aucune option permettant de renommer un utilisateur. Pour plus d'informations sur les autorisations requises pour renommer un utilisateur, consultez Autorisations requises pour accéder aux autres ressources IAM.

Lorsque vous modifiez le nom ou le chemin d'accès d'un utilisateur, voici ce qui suit se produit :

- Toutes les politiques attachées à l'utilisateur restent dans l'utilisateur sous le nouveau nom.
- L'utilisateur reste dans les mêmes groupes d'utilisateurs sous le nouveau nom.
- L'ID unique de l'utilisateur demeure le même. Pour plus d'informations sur les ID uniques, consultez Identifiants uniques.
- Toutes les ressources ou politiques de rôle qui font référence à l'utilisateur en tant que principal (l'utilisateur se voit accorder l'accès) sont mises à jour automatiquement pour utiliser le nouveau nom ou chemin. Par exemple, toutes les politiques basées sur des files d'attente dans Amazon SQS ou les politiques basées sur des ressources dans Amazon S3 sont mises à jour automatiquement pour utiliser le nouveau nom ou chemin.

IAM ne met pas automatiquement à jour les politiques qui font référence à l'utilisateur en tant que ressource de manière à utiliser le nouveau nom ou chemin ; vous devez le faire manuellement. Par exemple, imaginons qu'une politique est attachée à l'utilisateur `Richard` et qu'elle lui permet de gérer ses informations d'identification de sécurité. Si un administrateur renomme `Richard` en `Rich`, administrateur doit également mettre à jour cette politique pour changer la ressource de :

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Richard
```

à :

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Rich
```

C'est également vrai si un administrateur modifie le chemin : il doit également mettre à jour la politique pour refléter le nouveau chemin de l'utilisateur.

Pour renommer un utilisateur

- AWS CLI : aws iam update-user
- AWS API : UpdateUser

Suppression d'un utilisateur IAM

Vous pouvez supprimer un utilisateur IAM de votre entreprise Compte AWS s'il quitte votre entreprise. Si l'utilisateur est absent temporairement, vous pouvez désactiver l'accès de l'utilisateur au lieu de le supprimer du compte comme décrit dans Désactivation d'un utilisateur IAM.

Rubriques

- Suppression d'un utilisateur IAM (console)
- Suppression d'un utilisateur IAM (AWS CLI)

Suppression d'un utilisateur IAM (console)

Lorsque vous utilisez le AWS Management Console pour supprimer un utilisateur IAM, IAM supprime automatiquement les informations suivantes pour vous :

- L'utilisateur
- Toute appartenance à un groupe d'utilisateurs, ce qui signifie que l'utilisateur est supprimé de tous les groupes d'utilisateurs IAM dont il était membre
- Tous les mots de passe associés à utilisateur
- Toutes les clés d'accès appartenant à l'utilisateur
- Toutes les politiques en ligne intégrées à l'utilisateur (politiques qui sont appliquées à un utilisateur via les autorisations de groupe d'utilisateurs ne sont pas affectées)

> **Note**
>
> IAM supprime toutes les politiques gérées attachées à l'utilisateur lorsque vous supprimez l'utilisateur, mais ne supprime pas les politiques gérées.

- Tous les dispositifs MFA associés

Pour supprimer un utilisateur IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, sélectionnez Users (Utilisateurs), puis cochez la case en regard du nom d'utilisateur que vous souhaitez supprimer, pas le nom ou la ligne elle-même.
3. En haut de la page, sélectionnez Delete (Supprimer).
4. Dans la boîte de dialogue de confirmation, saisissez le nom d'utilisateur dans le champ de saisie de texte pour confirmer la suppression de l'utilisateur. Sélectionnez Supprimer.

Suppression d'un utilisateur IAM (AWS CLI)

Contrairement au AWS Management Console, lorsque vous supprimez un utilisateur avec le AWS CLI, vous devez supprimer les éléments attachés à l'utilisateur manuellement. La procédure suivante illustre ce processus.

Pour supprimer un utilisateur de votre compte (AWS CLI)

1. Supprimez le mot de passe de l'utilisateur, s'il en a un.

   `aws iam delete-login-profile`
2. Supprimez les clés d'accès de l'utilisateur, si l'utilisateur en possède une.

   `aws iam list-access-keys` (pour répertorier les clés d'accès de l'utilisateur) et `aws iam delete-access-key`
3. Supprimez le certificat de signature de l'utilisateur. Notez que la suppression d'une information d'identification de sécurité est définitive et que vous ne pourrez plus récupérer celle-ci.

   `aws iam list-signing-certificates` (pour répertorier les certificats de signature de l'utilisateur) et `aws iam delete-signing-certificate`

4. Supprimez la clé publique SSH de l'utilisateur, s'il en a une.

   `aws iam list-ssh-public-keys` (pour répertorier les clés publiques SSH de l'utilisateur) et `aws iam delete-ssh-public-key`

5. Supprimez les informations d'identification Git.

   `aws iam list-service-specific-credentials` (pour répertorier les informations d'identification git de l'utilisateur) et `aws iam delete-service-specific-credential`

6. Désactivez l'authentification multifacteur (MFA) de l'utilisateur, s'il en a une.

   `aws iam list-mfa-devices` (pour répertorier les dispositifs MFA de l'utilisateur), `aws iam deactivate-mfa-device` (pour désactiver le dispositif), et `aws iam delete-virtual-mfa-device` (pour supprimer définitivement un dispositif MFA virtuel)

7. Supprimez les politiques en ligne de l'utilisateur.

   `aws iam list-user-policies` (pour répertorier les politiques en ligne pour l'utilisateur) et `aws iam delete-user-policy` (pour supprimer la politique)

8. Détachez toutes les politiques gérées attachées à l'utilisateur.

   `aws iam list-attached-user-policies` (pour répertorier les politiques gérées attachées à l'utilisateur) et `aws iam detach-user-policy` (pour détacher la politique)

9. Supprimez l'utilisateur à partir de tous les groupes d'utilisateurs.

   `aws iam list-groups-for-user` (pour répertorier les groupes d'utilisateurs auxquels l'utilisateur appartient) et `aws iam remove-user-from-group`

10. Supprimez l'utilisateur.

    `aws iam delete-user`

Désactivation d'un utilisateur IAM

Vous pouvez désactiver un utilisateur IAM pendant qu'il est temporairement absent de votre entreprise. Vous pouvez laisser leurs informations d'identification d'utilisateur IAM en place tout en bloquant leur AWS accès.

Pour désactiver un utilisateur, créez et attachez une politique pour lui refuser l'accès à AWS. Vous pourrez rétablir l'accès de l'utilisateur ultérieurement.

Voici deux exemples de politiques de refus que vous pouvez attacher à un utilisateur pour lui refuser l'accès.

La politique suivante n'inclut pas de limite de temps. Vous devez supprimer la politique pour rétablir l'accès de l'utilisateur.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*"
      }
   ]
}
```

La politique suivante inclut une condition qui commence la politique le 24 décembre 2024 à 23 h 59 (UTC) et la termine le 28 février 2025 à 23 h 59 (UTC).

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
          "DateGreaterThan": {"aws:CurrentTime": "2024-12-24T23:59:59Z"},
          "DateLessThan": {"aws:CurrentTime": "2025-02-28T23:59:59Z"}
          }
       }
   ]
}
```

Modification des autorisations pour un utilisateur IAM

Vous pouvez modifier les autorisations d'un utilisateur IAM de votre site en Compte AWS modifiant son appartenance à un groupe, en copiant les autorisations d'un utilisateur existant, en attachant des politiques directement à un utilisateur ou en définissant une limite d'autorisation. Une limite

d'autorisations contrôle les autorisations maximum dont un utilisateur peut disposer. Les limites d'autorisations constituent une AWS fonctionnalité avancée.

Pour plus d'informations sur les autorisations requises pour modifier les autorisations d'un utilisateur, consultez Autorisations requises pour accéder aux autres ressources IAM.

Rubriques

- Afficher l'accès des utilisateurs
- Générer une politique basée sur l'activité d'accès d'un utilisateur
- Ajout d'autorisations à un utilisateur (console)
- Modification des autorisations pour un utilisateur (console)
- Suppression d'une politique d'autorisations d'un utilisateur (console)
- Suppression de la limite d'autorisations d'un utilisateur (console)
- Ajouter et supprimer les autorisations (AWS CLI ou AWS API) d'un utilisateur

Afficher l'accès des utilisateurs

Avant de modifier les autorisations d'un utilisateur, vous devez passer en revue ses activités récentes au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Générer une politique basée sur l'activité d'accès d'un utilisateur

Vous pouvez parfois octroyer plus d'autorisations à une entité IAM (utilisateur ou rôle) qu'elle n'en a besoin. Pour affiner les autorisations que vous octroyez, vous pouvez générer une politique IAM basée sur l'activité d'accès d'une entité. IAM Access Analyzer examine vos AWS CloudTrail journaux et génère un modèle de politique contenant les autorisations qui ont été utilisées par l'entité dans la plage de dates que vous avez spécifiée. Vous pouvez utiliser le modèle pour créer une politique gérée avec des autorisations affinées, puis l'attacher à l'entité IAM. Ainsi, vous accordez uniquement les autorisations dont l'utilisateur ou le rôle a besoin pour interagir avec les AWS ressources correspondant à votre cas d'utilisation spécifique. Pour en savoir plus, veuillez consulter la section Générer des politiques basées sur l'activité d'accès.

Ajout d'autorisations à un utilisateur (console)

IAM propose trois méthodes pour ajouter des politiques d'autorisations à un utilisateur :

- Add user to group (Ajouter un utilisateur à un groupe) : faites de l'utilisateur un membre d'un groupe. Les politiques du groupe sont attachées à l'utilisateur.
- Copy permissions from existing user (Copier les autorisations d'un utilisateur existant) : copiez toutes les appartenances à un groupe, toutes les politiques gérées attachées, les politiques en ligne et toutes les limites d'autorisations existantes d'un utilisateur source.
- Attach policies directly to user (Attacher directement des politiques à l'utilisateur) : attachez directement une politique gérée à l'utilisateur. Pour faciliter la gestion des autorisations, affectez vos stratégies à un groupe, puis faites de ces utilisateurs des membres des groupes appropriés.

> **Important**
>
> Si l'utilisateur dispose d'une limite d'autorisations, alors vous ne pouvez pas ajouter à un utilisateur plus d'autorisations que le permet la limite.

Ajout d'autorisations en ajoutant l'utilisateur à un groupe

L'ajout d'un utilisateur à un groupe affecte immédiatement l'utilisateur.

Pour ajouter des autorisations à un utilisateur en ajoutant celui-ci à un groupe

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Vérifiez l'appartenance actuelle des utilisateurs au groupe dans la colonne Groupes de la console. Au besoin, ajoutez la colonne à la table des utilisateurs en procédant comme suit :
    1. Au-dessus de la table à l'extrême droite, choisissez le symbole des paramètres ( ).
    2. Dans la boîte de dialogue Manage Columns (Gérer les colonnes), sélectionnez la colonne Groupes. Sinon, vous pouvez également désactiver la case à cocher des en-têtes de colonnes que vous ne souhaitez pas voir s'afficher dans la table des utilisateurs.
    3. Choisissez Fermer pour revenir à la liste des utilisateurs.

La colonne Groupes vous indique à quels groupes les utilisateurs appartiennent. La colonne inclut jusqu'à deux noms de groupes. Si l'utilisateur est membre d'au moins trois groupes, les deux premiers sont affichés (par ordre alphabétique) et le nombre d'appartenances supplémentaires à un groupe est inclus. Par exemple, si l'utilisateur fait partie du Groupe A, du Groupe B, du Groupe C et du Groupe D, le champ contient la valeur Group A, Group B + 2 more. Pour afficher le nombre total de groupes auxquels l'utilisateur appartient, vous pouvez ajouter la colonne Group count (Nombre de groupes) à la table de l'utilisateur.

4. Choisissez le nom de l'utilisateur dont vous souhaitez modifier les autorisations.
5. Choisissez l'onglet Autorisations, puis Add permissions (Ajouter des autorisations). Choisissez Add user to group (ajouter un utilisateur au groupe).
6. Activez la case à cocher pour chaque groupe que l'utilisateur doit rejoindre. La liste affiche le nom de chaque groupe et les politiques que l'utilisateur reçoit s'il devient membre de ce groupe.
7. (Facultatif) Outre la sélection de groupes existants, vous pouvez choisir Créer un groupe pour définir un nouveau groupe :

   a. Dans le nouvel onglet, pour Nom du groupe d'utilisateurs, saisissez le nom de votre nouveau groupe.

      > **Note**
      >
      > Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d'informations, consultez IAM et quotas AWS STS. Les noms de groupe peuvent combiner jusqu'à 128 lettres, chiffres et caractères suivants : plus (+), égal (=), virgule (,), point (.), arobase (@) et tiret (-). Les noms doivent être uniques dans un compte. Ils ne sont pas sensibles à la casse. Par exemple, vous ne pouvez pas créer deux groupes nommés TESTGROUP et testgroup.

   b. Activez une ou plusieurs cases à cocher pour les politiques gérées que vous souhaitez attacher au groupe. Vous pouvez également créer une politique gérée en choisissant Créer une politique. Le cas échéant, revenez dans la fenêtre ou l'onglet du navigateur une fois la nouvelle politique créée. Choisissez Refresh (Actualiser), puis choisissez la nouvelle politique à attacher à votre groupe. Pour plus d'informations, consultez Création de politiques IAM.

   c. Choisissez Créer un groupe d'utilisateurs.

d. Revenez à l'onglet initial, actualisez votre liste de groupes. Puis cochez la case correspondant à votre nouveau groupe.

8. Choisissez Suivant pour afficher la liste des membres du groupe à ajouter à l'utilisateur. Choisissez ensuite Add permissions (Ajouter des autorisations).

Ajout d'autorisations en copiant celles d'un autre utilisateur

La copie d'autorisations affecte immédiatement l'utilisateur.

Pour ajouter des autorisations à un utilisateur en copiant celles d'un autre utilisateur

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Utilisateurs, choisissez le nom de l'utilisateur dont vous souhaitez modifier les autorisations, puis choisissez l'onglet Autorisations.
3. Choisissez Add permissions (Ajouter des autorisations), puis choisissez Copy permissions from existing user (Copier des autorisations d'un utilisateur existant). La liste affiche les utilisateurs disponibles ainsi que leur appartenance à un groupe et les politiques attachées. Si la liste complète des groupes ou des politiques ne tient pas sur une seule ligne, vous pouvez choisir le lien pour et ***n***de plus. Un nouvel onglet de navigateur s'affiche avec une liste complète des politiques (onglet Autorisations) et des groupes (onglet Groupes).
4. Sélectionnez le bouton radio en regard de l'utilisateur dont vous voulez copier les autorisations.
5. Choisissez Suivant pour afficher la liste des modifications apportées à l'utilisateur. Choisissez ensuite Add permissions (Ajouter des autorisations).

Ajout d'autorisations en attachant les politiques directement à l'utilisateur

L'attachement de politiques affecte immédiatement l'utilisateur.

Pour ajouter des autorisations à un utilisateur en attachant directement les politiques gérées

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Utilisateurs, choisissez le nom de l'utilisateur dont vous souhaitez modifier les autorisations, puis choisissez l'onglet Autorisations.
3. Choisissez Ajouter des autorisations, puis Attacher directement des stratégies.

4. Cochez une ou plusieurs cases pour les politiques gérées que vous souhaitez attacher à l'utilisateur. Vous pouvez également créer une politique gérée en choisissant Créer une politique. Dans ce cas, revenez dans l'onglet ou la fenêtre du navigateur une fois la nouvelle politique créée. Choisissez Refresh (Actualiser), puis cochez la case en regard de la nouvelle politique pour l'attacher à votre utilisateur. Pour plus d'informations, consultez Création de politiques IAM.
5. Choisissez Suivant pour afficher la liste des politiques attachées à l'utilisateur. Choisissez ensuite Add permissions (Ajouter des autorisations).

Définition de la limite d'autorisations pour un utilisateur

La définition d'une limite d'autorisations affecte immédiatement l'utilisateur.

Pour définir la limite d'autorisations pour un utilisateur

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom de l'utilisateur dont vous souhaitez modifier la limite d'autorisations.
4. Choisissez l'onglet Permissions (Autorisations). Si nécessaire, ouvrez la section Limite d'autorisations, puis choisissez Définir une limite d'autorisations.
5. Sélectionnez la politique que vous souhaitez utiliser pour la limite d'autorisations.
6. Choisissez Set boundary (Définir une limite).

Modification des autorisations pour un utilisateur (console)

IAM vous permet de modifier les autorisations associées à un utilisateur de la manière suivante :

- Edit a permissions policy (Modifier une politique d'autorisations) : modifiez la politique en ligne d'un utilisateur, la politique en ligne du groupe de l'utilisateur ou une politique gérée attachée directement à l'utilisateur ou à partir d'un groupe. Si l'utilisateur dispose d'une limite d'autorisations, alors vous ne pouvez pas fournir plus d'autorisations que le permet la politique utilisée comme limite d'autorisations de l'utilisateur.
- Changing the permissions boundary (Modification de la limite d'autorisations) : modifiez la politique utilisée comme limite d'autorisations pour l'utilisateur. Cette action peut développer ou limiter les autorisations maximum dont dispose un utilisateur.

Modification d'une politique d'autorisations attachée à un utilisateur

La modification d'autorisations affecte immédiatement l'utilisateur.

Pour modifier les politiques gérées attachées d'un utilisateur

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom de l'utilisateur dont vous souhaitez modifier la politique d'autorisations.
4. Choisissez l'onglet Permissions (Autorisations). Si nécessaire, ouvrez la section Permissions policies (Politiques d'autorisations).
5. Choisissez le nom de la politique que vous souhaitez modifier pour en afficher les détails. Choisissez l'onglet Utilisation de la politique pour afficher d'autres entités qui pourraient être affectées par la modification de la politique.
6. Choisissez l'onglet Autorisations et examinez les autorisations accordées par la politique. Puis choisissez Modifier la politique.
7. Modifiez la politique et résolvez les recommandations sur la validation des politiques. Pour plus d'informations, consultez Modification de politiques IAM.
8. Choisissez Examiner une politique, examinez le récapitulatif de la politique, puis choisissez Enregistrer les modifications.

Modification de la limite d'autorisations pour un utilisateur

La modification d'une limite d'autorisations affecte immédiatement l'utilisateur.

Pour modifier la politique utilisée afin de définir la limite d'autorisations pour un utilisateur

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom de l'utilisateur dont vous souhaitez modifier la limite d'autorisations.
4. Choisissez l'onglet Permissions (Autorisations). Si nécessaire, ouvrez la section Permissions boundary (Limite d'autorisations), puis choisissez Change boundary (Modifier une limite).
5. Sélectionnez la politique que vous souhaitez utiliser pour la limite d'autorisations.

6. Choisissez Set boundary (Définir une limite).

Suppression d'une politique d'autorisations d'un utilisateur (console)

La suppression d'une politique affecte immédiatement l'utilisateur.

Pour supprimer les autorisations des utilisateurs IAM, procédez comme suit :

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom de l'utilisateur dont vous souhaitez supprimer la limite d'autorisations.
4. Choisissez l’onglet Permissions (Autorisations).
5. Si vous souhaitez supprimer les autorisations en supprimant une politique existante, affichez le type pour comprendre comment l'utilisateur obtient cette politique avant de choisir Supprimer pour supprimer celle-ci :

   - Si la politique s'applique en raison de l'appartenance à un groupe, choisissez Supprimer pour supprimer l'utilisateur du groupe. N'oubliez pas que vous pouvez avoir plusieurs politiques attachées à un seul groupe. Si vous supprimez un utilisateur d'un groupe, il perd l'accès à toutes les politiques qu'il reçoit par le biais de cette appartenance au groupe.
   - Si la politique est une politique gérée attachée directement à l'utilisateur, choisissez Supprimer pour détacher la politique de l'utilisateur. Cela n'affecte pas la politique elle-même ni aucune autre entité à laquelle la politique pourrait être attachée.
   - S'il s'agit d'une politique en ligne intégrée, sélectionnez X pour supprimer la politique d'IAM. Les politiques en ligne attachées directement à un utilisateur existent uniquement sur cet utilisateur.

Suppression de la limite d'autorisations d'un utilisateur (console)

La suppression d'une limite d'autorisations affecte immédiatement l'utilisateur.

Pour supprimer la limite d'autorisations d'un utilisateur

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.

3. Choisissez le nom de l'utilisateur dont vous souhaitez supprimer la limite d'autorisations.
4. Choisissez l’onglet Permissions (Autorisations). Si nécessaire, ouvrez la section Permissions boundary (Limite d'autorisations), puis choisissez Remove boundary (Supprimer une limite).
5. Choisissez Supprimer la limite pour confirmer la suppression de la limite d'autorisations.

Ajouter et supprimer les autorisations (AWS CLI ou AWS API) d'un utilisateur

Pour ajouter ou supprimer des autorisations par programme, vous devez ajouter ou supprimer l'appartenance au groupe, attacher ou détacher les politiques gérées, ou ajouter ou supprimer les politiques en ligne. Pour plus d’informations, consultez les rubriques suivantes :

- Ajout et suppression d'utilisateurs dans un groupe IAM
- Ajout et suppression d'autorisations basées sur l'identité IAM

Gestion des mots de passe utilisateur dans AWS

Vous pouvez gérer les mots de passe des utilisateurs IAM de votre compte. Les utilisateurs IAM ont besoin de mots de passe pour accéder au AWS Management Console. Les utilisateurs n'ont pas besoin de mots de passe pour accéder aux AWS ressources par programmation à l' AWS CLI aide des outils pour Windows PowerShell, des AWS SDK ou des API. Pour ces environnements, vous avez la possibilité d'attribuer des clés d'accès aux utilisateurs IAM. Cependant, il existe d'autres alternatives plus sécurisées aux clés d'accès que nous vous recommandons d'envisager d'abord. Pour plus d’informations, consultez AWS informations d'identification de sécurité.

Table des matières

- Définition d'une politique de mot de passe du compte pour les utilisateurs IAM
- Gestion des mots de passe des utilisateurs IAM
- Autorisation des utilisateurs IAM à modifier leurs propres mots de passe
- Modification par l'utilisateur IAM de son propre mot de passe

Définition d'une politique de mot de passe du compte pour les utilisateurs IAM

Vous pouvez définir une politique de mot de passe personnalisée Compte AWS pour définir les exigences de complexité et les périodes de rotation obligatoires pour les mots de passe de vos

utilisateurs IAM. Si vous ne définissez pas de politique de mot de passe personnalisée, les mots de passe des utilisateurs IAM doivent respecter la politique de AWS mot de passe par défaut. Pour plus d'informations, consultez Options de politique de mot de passe personnalisé.

Rubriques

- Règles relatives à la définition d'une politique de mot de passe
- Autorisations requises pour définir une politique de mot de passe
- Politique de mot de passe par défaut
- Options de politique de mot de passe personnalisé
- Définition d'une politique de mot de passe (console)
- Définition d'une politique de mot de passe (AWS CLI)
- Définition d'une politique de mot de passe (AWS API)

Règles relatives à la définition d'une politique de mot de passe

La politique de mot de passe IAM ne s'applique pas au Utilisateur racine d'un compte AWS mot de passe ou aux clés d'accès utilisateur IAM. Si un mot de passe expire, l'utilisateur IAM ne peut pas se connecter AWS Management Console mais peut continuer à utiliser ses clés d'accès.

Lorsque vous créez ou modifiez une politique de mot de passe, la plupart de ses paramètres sont appliqués la fois suivante où vos utilisateurs modifient leurs mots de passe. Toutefois, certains des paramètres sont appliqués immédiatement. Par exemple :

- Lorsque les exigences de longueur minimale et de type de caractères sont modifiés, les nouveaux paramètres sont appliqués à la prochaine modification des mots de passe. Les utilisateurs ne sont pas contraints à modifier leurs mots de passe, même si ceux-ci ne respectent pas la politique de mot de passe modifiée.
- Lorsque vous définissez une période d'expiration de mot de passe, celle-ci est appliquée immédiatement. Par exemple, supposons que vous définissez une période d'expiration de mot de passe de 90 jours. Dans ce cas, le mot de passe expire pour tous les utilisateurs IAM dont le mot de passe existant date de plus de 90 jours. Ces utilisateurs sont tenus de modifier leur mot de passe la première fois qu'ils se connectent.

Vous ne pouvez pas créer de « politique de verrouillage » pour empêcher l'accès d'un utilisateur à un compte après un nombre défini de tentatives de connexion ayant échoué. Pour renforcer votre sécurité, nous vous recommandons de combiner des politiques de mot de passe d'un niveau de

sécurité élevé à la Multi-Factor Authentication (MFA). Pour plus d'informations sur l'authentification MFA, consultez Utilisation de l'authentification multifactorielle (MFA) dans AWS.

Autorisations requises pour définir une politique de mot de passe

Vous devez configurer les autorisations pour permettre à une entité IAM (utilisateur ou rôle) d'afficher ou de modifier sa politique de mot de passe de compte. Vous pouvez inclure les actions de politique de mot de passe suivantes dans une politique IAM :

- `iam:GetAccountPasswordPolicy` : permet à l'entité d'afficher la politique de mot de passe pour son compte
- `iam:DeleteAccountPasswordPolicy` : permet à l'entité de supprimer la politique de mot de passe personnalisée pour son compte et de revenir à la politique de mot de passe par défaut
- `iam:UpdateAccountPasswordPolicy` : permet à l'entité de créer ou de modifier la politique de mot de passe personnalisée pour son compte

La politique suivante permet un accès complet pour afficher et modifier la politique de mot de passe du compte. Pour apprendre à créer une politique IAM à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessPasswordPolicy",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:DeleteAccountPasswordPolicy",
                "iam:UpdateAccountPasswordPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour plus d'informations sur les autorisations requises par les utilisateurs IAM pour modifier leur propre mot de passe, veuillez consulter Autorisation des utilisateurs IAM à modifier leurs propres mots de passe.

Politique de mot de passe par défaut

Si un administrateur ne définit pas de politique de mot de passe personnalisée, les mots de passe des utilisateurs IAM doivent respecter la politique de AWS mot de passe par défaut.

La politique de mot de passe par défaut exige les conditions suivantes :

- Longueur minimale du mot de passe de 8 caractères et longueur maximale de 128 caractères
- Au minimum trois des types de caractères suivants : majuscules, minuscules, chiffres et les caractères non alphanumériques (`! @ # $ % ^ & * ( ) _ + - = [ ] { } | '`)
- Ne pas être identique à votre Compte AWS nom ou à votre adresse e-mail
- Mot de passe sans expiration

Options de politique de mot de passe personnalisé

Lorsque vous configurez une politique de mot de passe personnalisée pour votre compte, vous pouvez spécifier les conditions suivantes :

- Password minimum length (Longueur minimale du mot de passe) : vous pouvez spécifier une longueur minimale de 6 caractères et une longueur maximale de 128 caractères.
- Password strength (Niveau de sécurité du mot de passe) : vous pouvez cocher l'une des cases suivantes pour définir le niveau de sécurité de vos mots de passe utilisateur IAM :
  - Exiger au moins une lettre majuscule de l'alphabet latin (A–Z)
  - Exiger au moins une lettre minuscule de l'alphabet latin (a–z)
  - Nécessite au moins un chiffre
  - Exiger au moins un caractère non alphanumérique `! @ # $ % ^ & * ( ) _ + - = [ ] { } | '`
- Turn on password expiration (Activer l'expiration du mot de passe) : vous pouvez sélectionner et spécifier une durée minimale de 1 jour et une durée maximale de 1 095 jours pendant laquelle les mots de passe utilisateur IAM sont valides après leur définition. Par exemple, si vous spécifiez un délai d'expiration de 90 jours, cela a un impact immédiat sur tous vos utilisateurs. Pour les utilisateurs dont le mot de passe date de plus de 90 jours, ils doivent définir un nouveau mot de passe lorsqu'ils se connectent à la console après la modification. Les utilisateurs dont le mot de passe est vieux de 75 à 89 jours reçoivent un AWS Management Console avertissement concernant l'expiration de leur mot de passe. Les utilisateurs IAM peuvent modifier leur mot de passe à tout moment s'ils en ont l'autorisation. Lorsqu'ils définissent un nouveau mot de passe, la

date d’expiration est réinitialisée. Un utilisateur IAM ne peut avoir qu'un mot de passe valide à la fois.

- L'expiration du mot de passe nécessite une réinitialisation par l'administrateur : sélectionnez cette option pour empêcher les utilisateurs IAM de l'utiliser AWS Management Console pour mettre à jour leur propre mot de passe après son expiration. Avant de sélectionner cette option, vérifiez que votre Compte AWS a plusieurs utilisateurs disposant des autorisations d'administrateur nécessaires pour réinitialiser les mots de passe utilisateur IAM. Les administrateurs disposant de l'autorisation `iam:UpdateLoginProfile` peuvent réinitialiser les mots de passe des utilisateurs IAM. Les utilisateurs IAM disposant de l'autorisation `iam:ChangePassword` et de clés d'accès actives peuvent réinitialiser leur propre mot de passe de console utilisateur IAM de manière programmatique. Si vous décochez cette case, les utilisateurs IAM dont les mots de passe ont expiré doivent tout de même définir un nouveau mot de passe avant de pouvoir accéder à la AWS Management Console.
- Allow users to change their own password (Autoriser les utilisateurs à modifier leur propre mot de passe) : vous pouvez permettre à tous les utilisateurs IAM de votre compte d'utiliser la console IAM pour modifier leur mot de passe. Cela permet aux utilisateurs d'accéder à l'action `iam:ChangePassword` uniquement pour leur propre utilisateur et à l'action `iam:GetAccountPasswordPolicy`. Cette option n'associe aucune politique d'autorisations à chaque utilisateur. IAM applique plutôt les autorisations au niveau du compte pour tous les utilisateurs. Vous pouvez également n'autoriser que certains utilisateurs à gérer leurs propres mots de passe. Pour ce faire, décochez cette case. Pour plus d'informations sur l'utilisation de politiques visant à limiter les utilisateurs pouvant gérer les mots de passe, consultez la section Autorisation des utilisateurs IAM à modifier leurs propres mots de passe.
- Prevent password reuse (Empêcher la réutilisation d'un mot de passe) : vous pouvez empêcher les utilisateurs IAM de réutiliser un certain nombre de mots de passe précédents. Vous pouvez spécifier le nombre de mots de passe précédents qui ne peuvent pas être réutilisés, entre 1 et 24 mots de passe.

Définition d'une politique de mot de passe (console)

Vous pouvez utiliser le AWS Management Console pour créer, modifier ou supprimer une politique de mot de passe personnalisée.

Pour créer une politique de mot de passe personnalisée (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.

2. Dans le panneau de navigation, choisissez Paramètres du compte.
3. Dans la section Password policy (Politique de mot de passe), sélectionnez Edit (Modifier).
4. Choisissez Custom (Personnalisé) pour utiliser une politique de mot de passe personnalisée.
5. Sélectionnez les options que vous souhaitez appliquer à votre politique de mot de passe, puis sélectionnez Enregistrer les modifications.
6. Confirmez que vous souhaitez définir la politique de mot de passe personnalisée en sélectionnant Définir personnalisé.

Pour modifier une politique de mot de passe personnalisée (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Paramètres du compte.
3. Dans la section Password policy (Politique de mot de passe), sélectionnez Edit (Modifier).
4. Sélectionnez les options que vous souhaitez appliquer à votre politique de mot de passe, puis sélectionnez Enregistrer les modifications.
5. Confirmez que vous souhaitez définir la politique de mot de passe personnalisée en sélectionnant Définir personnalisé.

Pour supprimer une politique de mot de passe personnalisée (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Paramètres du compte.
3. Dans la section Password policy (Politique de mot de passe), sélectionnez Edit (Modifier).
4. Choisissez IAM default (IAM par défaut) pour supprimer la politique de mot de passe personnalisée, puis Save changes (Enregistrer les modifications).
5. Confirmez que vous souhaitez définir la politique de mot de passe IAM par défaut en sélectionnant Définir personnalisé.

Définition d'une politique de mot de passe (AWS CLI)

Vous pouvez utiliser le AWS Command Line Interface pour définir une politique de mot de passe.

Pour gérer la politique de mot de passe de compte personnalisée à partir du AWS CLI

Exécutez les commandes suivantes :

- Pour créer ou modifier la politique de mot de passe personnalisée : `aws iam update-account-password-policy`
- Pour afficher la politique de mot de passe : `aws iam get-account-password-policy`
- Pour supprimer la politique de mot de passe personnalisée : `aws iam delete-account-password-policy`

Définition d'une politique de mot de passe (AWS API)

Vous pouvez utiliser les opérations de AWS l'API pour définir une politique de mot de passe.

Pour gérer la politique de mot de passe de compte personnalisée à partir de l' AWS API

Appelez les opérations suivantes :

- Pour créer ou modifier la politique de mot de passe personnalisée : `UpdateAccountPasswordPolicy`
- Pour afficher la politique de mot de passe : `GetAccountPasswordPolicy`
- Pour supprimer la politique de mot de passe personnalisée : `DeleteAccountPasswordPolicy`

Gestion des mots de passe des utilisateurs IAM

Les utilisateurs IAM qui utilisent les ressources AWS Management Console pour travailler avec AWS des ressources doivent disposer d'un mot de passe pour se connecter. Vous pouvez créer, modifier ou supprimer un mot de passe pour un utilisateur IAM dans votre compte AWS .

Une fois que vous avez attribué un mot de passe à un utilisateur, celui-ci peut se connecter à l' AWS Management Console aide de l'URL de connexion de votre compte, qui ressemble à ceci :

```
https://12-digit-AWS-account-ID or alias.signin.aws.amazon.com/console
```

Pour plus d'informations sur la manière dont les utilisateurs IAM se connectent au AWS Management Console, consultez la section Comment se connecter AWS dans le guide de l'Connexion à AWS utilisateur.

Même si vos utilisateurs ont leurs propres mots de passe, ils toujours besoin de votre autorisation pour accéder à vos ressources AWS . Par défaut, un utilisateur ne dispose d'aucune autorisation. Pour accorder à vos utilisateurs les autorisations dont ils ont besoin, vous leur attribuez des politiques

ou aux groupes dont ils font partie. Pour plus d'informations sur la création d'utilisateurs et de groupes, consultez Identités IAM (utilisateurs, groupes d'utilisateurs et rôles). Pour plus d'informations sur l'utilisation de stratégies pour l'octroi d'autorisations, consultez Modification des autorisations pour un utilisateur IAM.

Vous pouvez accorder aux utilisateurs l'autorisation de modifier leurs propres mots de passe. Pour plus d'informations, consultez Autorisation des utilisateurs IAM à modifier leurs propres mots de passe. Pour plus d'informations sur la façon dont les utilisateurs accèdent à la page de connexion à votre compte, consultez Connexion à AWS dans le Guide de l'utilisateur Connexion à AWS .

Rubriques

- Création, modification ou suppression d'un mot de passe utilisateur IAM (console)
- Création, modification ou suppression d'un mot de passe utilisateur IAM (AWS CLI)
- Création, modification ou suppression d'un mot de passe utilisateur IAM (AWS API)

Création, modification ou suppression d'un mot de passe utilisateur IAM (console)

Vous pouvez utiliser le AWS Management Console pour gérer les mots de passe de vos utilisateurs IAM.

Lorsque les utilisateurs quittent votre organisation ou n'ont plus besoin d'y AWS accéder, il est important de retrouver les informations d'identification qu'ils utilisaient et de s'assurer qu'elles ne sont plus opérationnelles. Idéalement, supprimez les informations d'identification qui ne sont plus requises. Il est toujours possible de les recréer ultérieurement, si nécessaire. Vous devez au moins modifier les informations d'identification afin que les anciens utilisateurs ne soient plus en mesure de s'en servir.

Pour ajouter un mot de passe pour un utilisateur IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom de l'utilisateur dont vous souhaitez créer le mot de passe.
4. Cliquez sur l'onglet Informations d'identification de sécurité, puis sous Connexion à la console, sélectionnez Activer l'accès à la console.
5. Dans Activer l'accès à la console, dans le champ Mot de passe de la console, choisissez si IAM doit générer un mot de passe ou créer un mot de passe personnalisé :

- Pour qu'IAM génère un mot de passe, sélectionnez Autogenerated password (Mot de passe généré automatiquement).
- Pour créer un mot de passe personnalisé, choisissez Custom password (Mot de passe personnalisé), puis tapez le mot de passe.

> **Note**
>
> Le mot de passe que vous créez doit respecter la politique de mot de passe du compte.

6. Pour obliger l'utilisateur à créer un nouveau mot de passe au moment de la connexion, sélectionnez L'utilisateur doit créer un nouveau mot de passe à la prochaine connexion. Choisissez ensuite Activer l'accès à la console.

> **Important**
>
> Si vous sélectionnez l'option L'utilisateur doit créer un nouveau mot de passe à la prochaine connexion, vérifiez que l'utilisateur est bien autorisé à modifier son mot de passe. Pour plus d'informations, consultez Autorisation des utilisateurs IAM à modifier leurs propres mots de passe.

7. Pour afficher le mot de passe afin de le partager avec l'utilisateur, choisissez Afficher dans la boîte de dialogue du mot de passe de la console.

> **Important**
>
> Pour des raisons de sécurité, vous ne pouvez pas accéder au mot de passe après avoir effectué cette étape, mais vous pouvez créer un nouveau mot de passe à tout moment.

Pour changer le mot de passe d'un utilisateur IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom de l'utilisateur dont vous souhaitez changer le mot de passe.

4. Cliquez sur l'onglet Informations d'identification de sécurité, puis sous Connexion à la console, sélectionnez Gérer l'accès à la console.
5. Dans Gérer l'accès à la console, choisissez Réinitialiser le mot de passe si ce n'est déjà fait. Si l'accès à la console est désactivé, aucun mot de passe n'est requis.
6. Pour accéder à la console, choisissez si IAM doit générer un mot de passe ou créer un mot de passe personnalisé :

   - Pour qu'IAM génère un mot de passe, sélectionnez Autogenerated password (Mot de passe généré automatiquement).
   - Pour créer un mot de passe personnalisé, choisissez Custom password (Mot de passe personnalisé), puis tapez le mot de passe.

   > **Note**
   >
   > Le mot de passe que vous créez doit respecter la politique de mot de passe du compte, si une politique est définie actuellement.

7. Pour obliger l'utilisateur à créer un nouveau mot de passe au moment de la connexion, sélectionnez L'utilisateur doit créer un nouveau mot de passe à la prochaine connexion.

   > **Important**
   >
   > Si vous sélectionnez l'option L'utilisateur doit créer un nouveau mot de passe à la prochaine connexion, vérifiez que l'utilisateur est bien autorisé à modifier son mot de passe. Pour plus d'informations, consultez Autorisation des utilisateurs IAM à modifier leurs propres mots de passe.

8. Pour révoquer les sessions de console actives de l'utilisateur, choisissez Révoquer les sessions de console actives. Choisissez ensuite Apply(Applisuer).

   Lorsque vous révoquez des sessions de console actives pour un utilisateur, IAM associe une nouvelle politique intégrée à l'utilisateur qui refuse toutes les autorisations pour toutes les actions. Il inclut une condition qui applique les restrictions uniquement si la session a été créée avant le moment où vous révoquez les autorisations, ainsi qu'environ 30 secondes dans le futur. Si l'utilisateur crée une nouvelle session après que vous ayez révoqué les autorisations, la politique de refus ne s'applique pas à cet utilisateur. Si un utilisateur révoque ses propres

sessions de console actives à l'aide de cette méthode, il sera immédiatement déconnecté du AWS Management Console.

> **Important**
>
> Pour révoquer avec succès les sessions de console actives d'un utilisateur, vous devez disposer de l'`PutUserPolicy`autorisation pour cet utilisateur. Cela vous permet d'associer la politique `AWSRevokeOlderSessions` intégrée à l'utilisateur.

9. Pour afficher le mot de passe afin de le partager avec l'utilisateur, choisissez Afficher dans la boîte de dialogue du mot de passe de la console.

> **Important**
>
> Pour des raisons de sécurité, vous ne pouvez pas accéder au mot de passe après avoir effectué cette étape, mais vous pouvez créer un nouveau mot de passe à tout moment.

Supprimer (désactiver) le mot de passe d'un utilisateur IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom de l'utilisateur dont vous souhaitez supprimer le mot de passe.
4. Cliquez sur l'onglet Informations d'identification de sécurité, puis sous Connexion à la console, sélectionnez Gérer l'accès à la console.
5. Dans Gérer l'accès à la console, choisissez Désactiver l'accès à la console si ce n'est déjà fait. Si l'accès à la console est désactivé, aucun mot de passe n'est requis.
6. Pour révoquer les sessions de console actives de l'utilisateur, choisissez Révoquer les sessions de console actives. Choisissez ensuite Désactiver l'accès.

> **Important**
>
> Pour révoquer avec succès les sessions de console actives d'un utilisateur, vous devez disposer de l'`PutUserPolicy`autorisation pour cet utilisateur. Cela vous permet d'associer la politique `AWSRevokeOlderSessions` intégrée à l'utilisateur.

Lorsque vous révoquez des sessions de console actives pour un utilisateur, IAM intègre une nouvelle politique intégrée à l'utilisateur IAM qui refuse toutes les autorisations pour toutes les actions. Il inclut une condition qui applique les restrictions uniquement si la session a été créée avant le moment où vous révoquez les autorisations, ainsi qu'environ 30 secondes dans le futur. Si l'utilisateur crée une nouvelle session après que vous ayez révoqué les autorisations, la politique de refus ne s'applique pas à cet utilisateur. Si un utilisateur révoque ses propres sessions de console actives à l'aide de cette méthode, il sera immédiatement déconnecté du AWS Management Console.

> **Important**
>
> Vous pouvez empêcher un utilisateur IAM d'accéder au AWS Management Console en supprimant son mot de passe. Cela les empêche de se connecter à l' AWS Management Console aide de leurs identifiants de connexion. Cela ne modifie pas ses autorisations ni ne l'empêche d'accéder à la console à l'aide d'un rôle endossé. Si l'utilisateur dispose de clés d'accès actives, elles continuent de fonctionner et autorisent l' AWS CLI accès via les Outils pour Windows PowerShell, l' AWS API ou l'Application AWS Console Mobile.

Création, modification ou suppression d'un mot de passe utilisateur IAM (AWS CLI)

Vous pouvez utiliser l' AWS CLI API pour gérer les mots de passe de vos utilisateurs IAM.

Pour créer un mot de passe (AWS CLI)

1. (Facultatif) Pour déterminer si un utilisateur possède un mot de passe, exécutez cette commande : aws iam get-login-profile
2. Pour créer un mot de passe, exécutez cette commande : aws iam create-login-profile

Pour modifier le mot de passe d'un utilisateur (AWS CLI)

1. (Facultatif) Pour déterminer si un utilisateur possède un mot de passe, exécutez cette commande : aws iam get-login-profile
2. Pour modifier un mot de passe, exécutez cette commande : aws iam update-login-profile

Pour supprimer (désactiver) le mot de passe d'un utilisateur (AWS CLI)

1. (Facultatif) Pour déterminer si un utilisateur possède un mot de passe, exécutez cette commande : aws iam get-login-profile
2. (Facultatif) Pour déterminer quand un mot de passe a été utilisé pour la dernière fois, exécutez cette commande : aws iam get-user
3. Pour supprimer un mot de passe, exécutez cette commande : aws iam delete-login-profile

> **Important**
>
> Lorsque vous supprimez le mot de passe d'un utilisateur, ce dernier ne peut plus se connecter à l'interface AWS Management Console. Si l'utilisateur dispose de clés d'accès actives, elles continuent de fonctionner et autorisent l'accès via les appels de fonction AWS CLI PowerShell, Outils pour Windows ou AWS API. Lorsque vous utilisez les AWS CLI outils pour Windows PowerShell ou l' AWS API pour supprimer un utilisateur de votre compte Compte AWS, vous devez d'abord supprimer le mot de passe à l'aide de cette opération. Pour plus d'informations, consultez Suppression d'un utilisateur IAM (AWS CLI).

Pour révoquer les sessions de console actives d'un utilisateur avant une heure spécifiée ()AWS CLI

1. Pour intégrer une politique en ligne qui révoque les sessions de console actives d'un utilisateur IAM avant une heure spécifiée, utilisez la politique en ligne suivante et exécutez cette commande : aws iam put-user-policy

   Cette politique intégrée refuse toutes les autorisations et inclut la clé de `lois : TokenIssue Heure` condition. Il révoque les sessions de console actives de l'utilisateur avant l'heure spécifiée dans l'`Condition`élément de la politique intégrée. Remplacez la valeur de la clé de `aws:TokenIssueTime` condition par votre propre valeur.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {
```

```
        "aws:TokenIssueTime": "2014-05-07T23:47:00Z"
      }
    }
  }
}
```

2. (Facultatif) Pour répertorier les noms des politiques intégrées à l'utilisateur IAM, exécutez cette commande : aws iam list-user-policies
3. (Facultatif) Pour afficher la politique intégrée nommée intégrée à l'utilisateur IAM, exécutez cette commande : aws iam get-user-policy

Création, modification ou suppression d'un mot de passe utilisateur IAM (AWS API)

Vous pouvez utiliser l' AWS API pour gérer les mots de passe de vos utilisateurs IAM.

Pour créer un mot de passe (AWS API)

1. (Facultatif) Pour déterminer si un utilisateur possède un mot de passe, appelez cette opération : GetLoginProfile
2. Pour créer un mot de passe, appelez cette opération : CreateLoginProfile

Pour modifier le mot de passe d'un utilisateur (AWS API)

1. (Facultatif) Pour déterminer si un utilisateur possède un mot de passe, appelez cette opération : GetLoginProfile
2. Pour modifier un mot de passe, procédez comme suit : UpdateLoginProfile

Pour supprimer (désactiver) le mot de passe d'un utilisateur (AWS API)

1. (Facultatif) Pour déterminer si un utilisateur possède un mot de passe, exécutez cette commande : GetLoginProfile
2. (Facultatif) Pour déterminer quand un mot de passe a été utilisé pour la dernière fois, exécutez cette commande : GetUser
3. Pour supprimer un mot de passe, exécutez cette commande : DeleteLoginProfile

> **Important**
>
> Lorsque vous supprimez le mot de passe d'un utilisateur, ce dernier ne peut plus se connecter à l'interface AWS Management Console. Si l'utilisateur dispose de clés d'accès actives, elles continuent de fonctionner et autorisent l'accès via les appels de fonction AWS CLI PowerShell, Outils pour Windows ou AWS API. Lorsque vous utilisez les AWS CLI outils pour Windows PowerShell ou l' AWS API pour supprimer un utilisateur de votre compte Compte AWS, vous devez d'abord supprimer le mot de passe à l'aide de cette opération. Pour plus d'informations, consultez Suppression d'un utilisateur IAM (AWS CLI).

Pour révoquer les sessions de console actives d'un utilisateur avant une heure spécifiée (AWS API)

1. Pour intégrer une politique en ligne qui révoque les sessions de console actives d'un utilisateur IAM avant une heure spécifiée, utilisez la stratégie en ligne suivante et exécutez cette commande : PutUserPolicy

   Cette politique intégrée refuse toutes les autorisations et inclut la clé de `lois : TokenIssue Heure` condition. Il révoque les sessions de console actives de l'utilisateur avant l'heure spécifiée dans l'`Condition`élément de la politique intégrée. Remplacez la valeur de la clé de `aws:TokenIssueTime` condition par votre propre valeur.

   ```
   {
     "Version": "2012-10-17",
     "Statement": {
       "Effect": "Deny",
       "Action": "*",
       "Resource": "*",
       "Condition": {
         "DateLessThan": {
           "aws:TokenIssueTime": "2014-05-07T23:47:00Z"
         }
       }
     }
   }
   ```

2. (Facultatif) Pour répertorier les noms des politiques intégrées à l'utilisateur IAM, exécutez cette commande : ListUserPolicies
3. (Facultatif) Pour afficher la politique intégrée nommée intégrée à l'utilisateur IAM, exécutez cette commande : GetUserPolicy

Autorisation des utilisateurs IAM à modifier leurs propres mots de passe

> **Note**
>
> Les utilisateurs dotés d'identités fédérées utiliseront le processus défini par leur fournisseur d'identité pour modifier leurs mots de passe. Il est recommandé d'obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires.

Vous pouvez accorder aux utilisateurs IAM l'autorisation de modifier leurs propres mots de passe lors de la connexion à la AWS Management Console. Vous pouvez effectuer cette opération de deux manières :

- Autorisez tous les utilisateurs IAM du compte à modifier leurs propres mots de passe.
- Autorisez uniquement les utilisateurs IAM sélectionnés à modifier leurs propres mots de passe. Dans ce scénario, vous désactivez l'option permettant à tous les utilisateurs de modifier leurs propres mots de passe et vous utilisez une politique IAM pour accorder des autorisations uniquement à certains utilisateurs. Cette approche permet à ces utilisateurs de modifier leurs propres mots de passe et éventuellement d'autres informations d'identification telles que leurs propres clés d'accès.

> **Important**
>
> Nous vous recommandons de définir une politique de mot de passe personnalisée qui oblige les utilisateurs IAM à créer des mots de passe d'un niveau de sécurité élevé.

Pour autoriser tous les utilisateurs IAM à modifier leurs propres mots de passe

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, cliquez sur Paramètres du compte.
3. Dans la section Password policy (Politique de mot de passe), sélectionnez Edit (Modifier).
4. Choisissez Custom (Personnalisé) pour utiliser une politique de mot de passe personnalisée.

5. Sélectionnez Allow users to change their own password (Autoriser les utilisateurs à modifier leur propre mot de passe), puis cliquez sur Save changes (Enregistrer les modifications). Cela permet à tous les utilisateurs du compte d'accéder à l'action `iam:ChangePassword` uniquement pour leur propre utilisateur et à l'action `iam:GetAccountPasswordPolicy`.
6. Fournissez aux utilisateurs les instructions suivantes pour modifier leurs mots de passe : Modification par l'utilisateur IAM de son propre mot de passe.

Pour plus d'informations sur les commandes AWS CLI, les outils pour Windows PowerShell et l'API que vous pouvez utiliser pour modifier la politique de mot de passe du compte (qui permet notamment à tous les utilisateurs de modifier leur propre mot de passe), consultezDéfinition d'une politique de mot de passe (AWS CLI).

Pour autoriser des utilisateurs IAM sélectionnés à modifier leurs propres mots de passe

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, cliquez sur Paramètres du compte.
3. Dans la section Paramètres du compte, vérifiez que l'option Autoriser les utilisateurs à modifier leur propre mot de passe n'est pas sélectionnée. Si cette case à cocher est activée, tous les utilisateurs peuvent modifier leurs propres mots de passe. (Reportez-vous à la procédure précédente.)
4. Créez les utilisateurs qui devraient être autorisés à modifier leurs propres mots de passe, s'il n'existe pas encore. Pour plus de détails, consultez Création d'un utilisateur IAM dans votre Compte AWS.
5. (Facultatif) Créez un groupe IAM pour les utilisateurs qui doivent être autorisés à modifier leurs mots de passe, puis ajoutez les utilisateurs de l'étape précédente au groupe. Pour plus de détails, consultez Gestion des groupes IAM.
6. Affectez la politique suivante au groupe. Pour plus d'informations, consultez Gestion des politiques IAM.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:GetAccountPasswordPolicy",
      "Resource": "*"
```

```
    },
    {
      "Effect": "Allow",
      "Action": "iam:ChangePassword",
      "Resource": "arn:aws:iam::*:user/${aws:username}"
    }
  ]
}
```

Cette politique donne accès à l'ChangePasswordaction, qui permet aux utilisateurs de modifier uniquement leurs propres mots de passe depuis la console AWS CLI, les outils pour Windows PowerShell ou l'API. Elle donne également accès à l'GetAccountPasswordPolicyaction, qui permet à l'utilisateur de consulter la politique de mot de passe actuelle ; cette autorisation est requise pour que l'utilisateur puisse consulter la politique de mot de passe du compte sur la page Modifier le mot de passe. L'utilisateur doit être autorisé à lire la politique de mot de passe actuelle pour s'assurer que le mot de passe satisfait les exigences de politique.

7. Fournissez aux utilisateurs les instructions suivantes pour modifier leurs mots de passe : Modification par l'utilisateur IAM de son propre mot de passe.

Pour plus d'informations

Pour plus d'informations sur la gestion des informations d'identification, consultez les rubriques suivantes :

- Autorisation des utilisateurs IAM à modifier leurs propres mots de passe
- Gestion des mots de passe utilisateur dans AWS
- Définition d'une politique de mot de passe du compte pour les utilisateurs IAM
- Gestion des politiques IAM
- Modification par l'utilisateur IAM de son propre mot de passe

Modification par l'utilisateur IAM de son propre mot de passe

Si vous avez été autorisé à modifier votre propre mot de passe utilisateur IAM, vous pouvez utiliser une page spéciale AWS Management Console à cet effet. Vous pouvez également utiliser l' AWS API AWS CLI or.

Rubriques

- Autorisations nécessaires
- Comment les utilisateurs IAM modifient leur mot de passe (console)
- Comment les utilisateurs IAM modifient leur propre mot de passe (AWS CLI ou AWS API)

Autorisations nécessaires

Pour modifier le mot de passe pour votre propre utilisateur IAM, vous devez disposer des autorisations de la politique suivante : AWS: permet aux utilisateurs IAM de modifier leur propre mot de passe de console sur la page des informations d'identification de sécurité.

Comment les utilisateurs IAM modifient leur mot de passe (console)

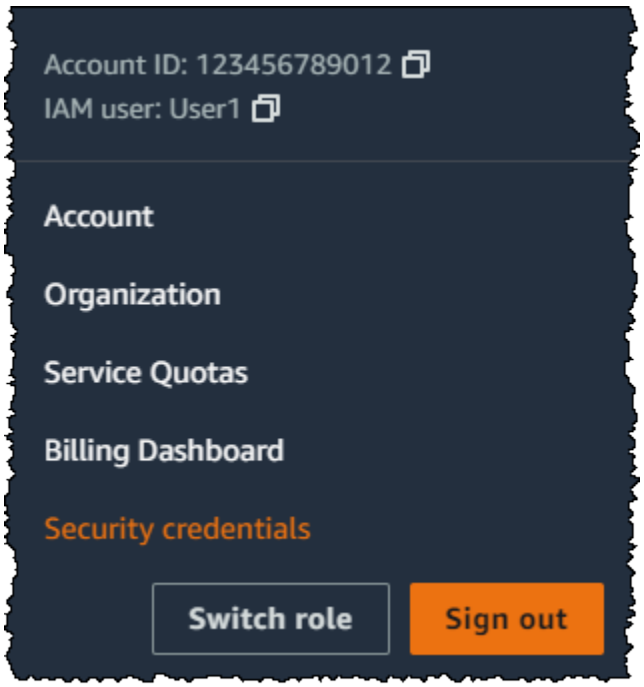
La procédure suivante décrit comment les utilisateurs IAM peuvent utiliser le AWS Management Console pour modifier leur propre mot de passe.

Pour modifier votre propre mot de passe utilisateur IAM (console)

1. Utilisez votre AWS identifiant ou alias de compte, votre nom d'utilisateur IAM et votre mot de passe pour vous connecter à la console IAM.

   > **Note**
   >
   > Pour votre commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser votre nom d'utilisateur IAM et les informations de votre compte. Si vous vous êtes déjà connecté en tant qu'utilisateur différent, sélectionnez Sign in to a different account (Se connecter à un compte différent) en bas de la page pour revenir à la page de connexion principale. À partir de là, vous pouvez saisir votre identifiant de AWS compte ou votre alias de compte pour être redirigé vers la page de connexion utilisateur IAM de votre compte.

   Pour obtenir votre Compte AWS identifiant, contactez votre administrateur.
2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).



3. Dans l'onglet Informations d'identification AWS IAM, sélectionnez Mettre à jour le mot de passe.
4. Dans le champ Current password (Mot de passe actuel), saisissez le mot de passe que vous utilisez actuellement. Entrez un nouveau mot de passe dans le champ New password (Nouveau mot de passe), puis confirmez-le dans le champ Confirm new password (Confirmer le nouveau mot de passe). Ensuite, choisissez l'option Mettre à jour le mot de passe.

> **Note**
>
> Le nouveau mot de passe doit répondre aux exigences de la politique de mot de passe du compte. Pour plus d'informations, consultez Définition d'une politique de mot de passe du compte pour les utilisateurs IAM.

Comment les utilisateurs IAM modifient leur propre mot de passe (AWS CLI ou AWS API)

La procédure suivante décrit comment les utilisateurs IAM peuvent utiliser l' AWS API AWS CLI or pour modifier leur propre mot de passe.

Pour modifier votre mot de passe IAM, utilisez ce qui suit :

- AWS CLI: `aws iam change-password`
- AWS API : `ChangePassword`

Gestion des clés d'accès pour les utilisateurs IAM

Follow us on Twitter

> **Important**
>
> Les bonnes pratiques consistent à utiliser des informations d'identification de sécurité temporaires (comme des rôles IAM) plutôt que de créer des informations d'identification à long terme comme des clés d'accès. Avant de créer des clés d'accès, passez en revue les alternatives aux clés d'accès à long terme.

Les clés d'accès sont les informations d'identification à long terme d'un utilisateur IAM ou du Utilisateur racine d'un compte AWS. Vous pouvez utiliser les clés d'accès pour signer des demandes programmatiques adressées à l' AWS API AWS CLI or (directement ou à l'aide du AWS SDK). Pour plus d'informations, consultez Signature des demandes AWS d'API.

Les clés d'accès se composent de deux parties : un ID de clé d'accès (par exemple, `AKIAIOSFODNN7EXAMPLE`) et une clé d'accès secrète (par exemple, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Vous devez utiliser à la fois l'ID de la clé d'accès et la clé d'accès secrète pour authentifier vos demandes.

Lorsque vous créez une paire de clé d'accès, enregistrez l'ID de clé d'accès et la clé d'accès secrète dans un emplacement sécurisé. La clé d'accès secrète est accessible uniquement au moment de sa création. Si vous perdez votre clé d'accès secrète, vous devez supprimer la clé d'accès et en créer une nouvelle. Pour en savoir plus, consultez Réinitialisation de mots de passe ou de clés d'accès perdus ou oubliés pour AWS.

Vous pouvez avoir un maximum de deux clés d'accès par utilisateur.

> **Important**
>
> Gérez vos clés d'accès en toute sécurité. Ne communiquez pas vos clés d'accès à des tiers non autorisés, même pour vous aider à trouver les identifiants de votre compte. En effet, vous lui accorderiez ainsi un accès permanent à votre compte.

Les rubriques suivantes détaillent les tâches de gestion associées aux clés d'accès.

Rubriques

- Autorisations requises pour gérer les clés d'accès
- Gestion des clés d'accès (console)
- Gestion des clés d'accès (AWS CLI)
- Gestion des clés d'accès (AWS API)
- Mise à jour des clés d'accès
- Sécurisation des clés d'accès
- Audit des clés d'accès

Autorisations requises pour gérer les clés d'accès

> **Note**
>
> `iam:TagUser` est une autorisation facultative permettant d'ajouter et de modifier des descriptions pour la clé d'accès. Pour plus d'informations, consultez Étiquette d'utilisateurs IAM.

Pour créer des clés d'accès pour votre propre utilisateur IAM, vous devez disposer des autorisations de la politique suivante :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Pour mettre à jour des clés d'accès pour votre propre utilisateur IAM, vous devez disposer des autorisations de la politique suivante :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:GetAccessKeyLastUsed",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Gestion des clés d'accès (console)

Vous pouvez utiliser le AWS Management Console pour gérer les clés d'accès d'un utilisateur IAM.

Pour créer, modifier ou supprimer vos propres clés d'accès (outil)

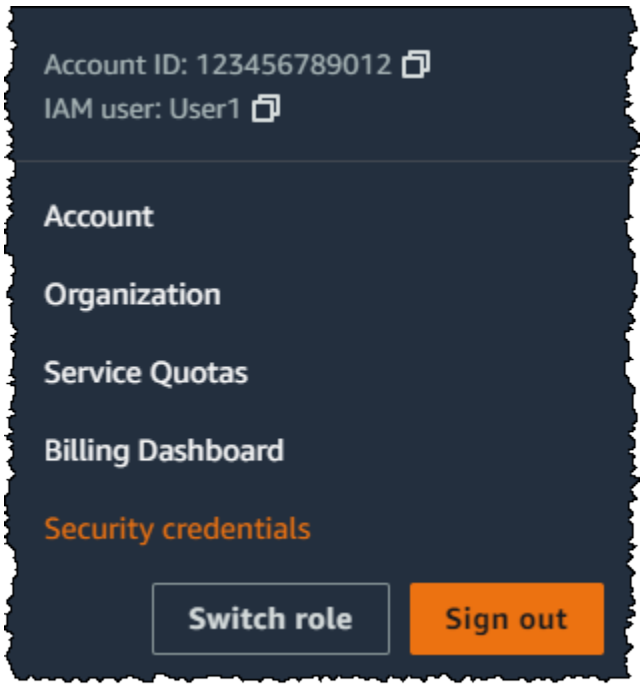
1. Utilisez votre AWS identifiant ou alias de compte, votre nom d'utilisateur IAM et votre mot de passe pour vous connecter à la console IAM.

   > **Note**
   >
   > Pour votre commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser votre nom d'utilisateur IAM et les informations de votre compte. Si vous vous êtes déjà connecté en tant qu'utilisateur différent, sélectionnez Sign in to a different account (Se connecter à un compte différent) en bas de la page pour revenir à la page de connexion principale. À partir de là, vous pouvez saisir votre identifiant de AWS

compte ou votre alias de compte pour être redirigé vers la page de connexion utilisateur IAM de votre compte.

Pour obtenir votre Compte AWS identifiant, contactez votre administrateur.

2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).



Effectuez l'une des actions suivantes :

Pour créer une clé d'accès

1. Dans la section Clés d'accès, choisissez Créer une clé d'accès. Si vous avez déjà deux clés d'accès, ce bouton est désactivé et vous devez supprimer une clé d'accès avant de pouvoir en créer une nouvelle.
2. Sur la page des bonnes pratiques et alternatives en matière de clés d'accès, choisissez votre cas d'utilisation pour découvrir les options supplémentaires qui peuvent vous aider à éviter de créer une clé d'accès à long terme. Si vous déterminez que votre cas d'utilisation nécessite toujours une clé d'accès, choisissez Other (Autre), puis Next (Suivant).
3. (Facultatif) Définissez une valeur de balise de description pour la clé d'accès. Cela permet d'ajouter une paire clé-valeur de balise à votre utilisateur IAM. Cela peut vous aider à identifier et

à mettre à jour les clés d'accès par la suite. La clé de balise est définie sur l'identifiant de la clé d'accès. La valeur de balise est définie selon la description de la clé d'accès que vous spécifiez. Lorsque vous avez terminé, sélectionnez Create access key (Créer la clé d'accès).

4. Sur la page Retrieve access keys (Récupérer les clés d'accès), choisissez Show (Afficher) (pour révéler la valeur de la clé d'accès secrète de votre utilisateur) ou Download .csv file (Télécharger le fichier .csv). C'est le seul moment où vous pouvez enregistrer votre clé d'accès secrète. Après avoir enregistré votre clé d'accès secrète dans un emplacement sécurisé, sélectionnez Done (Terminé).

Pour désactiver une clé d'accès

- Dans la section Access keys (Clés d'accès), recherchez la clé que vous souhaitez désactiver, puis choisissez Actions, puis Deactivate (Désactiver). À l'invite de confirmation, cliquez sur Deactivate (Désactiver). Une clé d'accès désactivée compte toujours dans votre limite de deux clés d'accès.

Pour activer une clé d'accès

- Dans la section Access keys (Clés d'accès), recherchez la clé que vous souhaitez activer, puis choisissez Actions, puis Activate (Activer).

Pour supprimer une clé d'accès dont vous n'avez plus besoin

- Dans la section Access keys (Clés d'accès), recherchez la clé que vous souhaitez suprimer, puis choisissez Actions, puis Delete (Supprimer). Suivez les instructions de la boîte de dialogue pour tout d'abord désactiver, puis confirmer la suppression. Nous vous recommandons de vérifier que la clé d'accès n'est plus utilisée avant de la supprimer définitivement.

Créer, modifier ou supprimer les clés d'accès d'un autre utilisateur IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom utilisateur dont vous souhaitez gérer les clés d'accès, puis sélectionnez l'onglet Informations d'identification de sécurité.
4. Dans la section Clés d'accès, procédez comme suit :

- Pour créer une clé d'accès, choisissez Créer une clé d'accès. Si le bouton est désactivé, vous devez supprimer l'une des clés existantes avant de pouvoir en créer une nouvelle. Sur la page Bonnes pratiques et alternatives en matière de clés d'accès, passez en revue les bonnes pratiques et alternatives. Choisissez votre cas d'utilisation pour découvrir les options supplémentaires qui peuvent vous aider à éviter de créer une clé d'accès à long terme. Si vous déterminez que votre cas d'utilisation nécessite toujours une clé d'accès, choisissez Other (Autre), puis Next (Suivant). Sur la page Retrieve access key page (Récupérer les clés d'accès), choisissez Show (Afficher) pour révéler la valeur de la clé d'accès secrète de votre utilisateur. Pour enregistrer l'ID de clé d'accès et la clé d'accès secrète dans un fichier `.csv`, dans un emplacement sécurisé sur votre ordinateur, sélectionnez le bouton Download .csv file (Télécharger un fichier .csv). Lorsque vous créez une clé d'accès pour votre utilisateur, cette paire de clés est activée par défaut et votre utilisateur peut immédiatement l'utiliser.
- Pour désactiver une clé d'accès active, choisissez Actions, puis Deactivate (Désactiver).
- Pour activer une clé d'accès inactive, choisissez Actions, puis Activate (Activer).
- Pour supprimer votre clé d'accès, choisissez Actions, puis Delete (Supprimer). Suivez les instructions de la boîte de dialogue pour tout d'abord désactiver, puis confirmer la suppression. AWS vous recommande de procéder comme suit : vous désactivez d'abord la clé et vérifiez qu'elle n'est plus utilisée. Lorsque vous utilisez le AWS Management Console, vous devez désactiver votre clé avant de la supprimer.

Pour répertorier les clés d'accès d'un utilisateur IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom utilisateur concerné, puis sélectionnez l'onglet Informations d'identification de sécurité. Dans la section Clés d'accès, vous pouvez consulter les clés d'accès de l'utilisateur ainsi que le statut de celles-ci.

> **Note**
>
> Seul l'ID de clé d'accès de l'utilisateur est visible. La clé d'accès secrète peut être récupérée uniquement au moment de la création de la clé.

Pour répertorier les ID de clés d'accès de plusieurs utilisateurs IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Au besoin, ajoutez la colonne ID de clé d'accès à la table des utilisateurs en procédant comme suit :

   a. Au-dessus de la table à l'extrême droite, choisissez l'icône des paramètres ( ).

   b. Dans Gérer les colonnes, sélectionnez ID de clé d'accès.

   c. Choisissez Fermer pour revenir à la liste des utilisateurs.
4. La colonne ID de clé d'accès présente chaque ID de clé d'accès, suivi par son état ; par exemple, 234782070278420732307623740 23 (Active) ou 220937402396702370248434203 27 (Inactive).

   Vous pouvez utiliser ces informations pour afficher et copier les clés d'accès des utilisateurs ayant une ou deux clés d'accès. La colonne affiche Aucun pour les utilisateurs sans clé d'accès.

   > **Note**
   >
   > Seul l'ID et l'état de la clé d'accès de l'utilisateur sont visibles. La clé d'accès secrète peut être récupérée uniquement au moment de la création de la clé.

Pour rechercher quel utilisateur IAM possède une clé d'accès spécifique (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Dans la zone de recherche, saisissez ou collez l'ID de la clé d'accès de l'utilisateur que vous recherchez.
4. Au besoin, ajoutez la colonne ID de clé d'accès à la table des utilisateurs en procédant comme suit :

a. Au-dessus de la table à l'extrême droite, choisissez l'icône des paramètres ( ).

b. Dans Gérer les colonnes, sélectionnez ID de clé d'accès.

c. Choisissez Fermer pour retourner la liste des utilisateurs et vérifier que l'utilisateur filtré est le propriétaire de la clé d'accès spécifiée.

Gestion des clés d'accès (AWS CLI)

Pour gérer les clés d'accès utilisateur IAM depuis le AWS CLI, exécutez les commandes suivantes.

- Pour créer une clé d'accès : `aws iam create-access-key`
- Pour activer ou désactiver une clé d'accès : `aws iam update-access-key`
- Pour répertorier les clés d'accès d'un utilisateur : `aws iam list-access-keys`
- Pour déterminer quand une clé d'accès a été utilisée pour la dernière fois : `aws iam get-access-key-last-used`
- Pour supprimer une clé d'accès : `aws iam delete-access-key`

Gestion des clés d'accès (AWS API)

Pour gérer les clés d'accès d'un utilisateur IAM depuis l' AWS API, effectuez les opérations suivantes.

- Pour créer une clé d'accès : `CreateAccessKey`
- Pour activer ou désactiver une clé d'accès : `UpdateAccessKey`
- Pour répertorier les clés d'accès d'un utilisateur : `ListAccessKeys`
- Pour déterminer quand une clé d'accès a été utilisée pour la dernière fois : `GetAccessKeyLastUsed`
- Pour supprimer une clé d'accès : `DeleteAccessKey`

Mise à jour des clés d'accès

Comme bonne pratique en matière de sécurité, nous vous recommandons de mettre à jour les clés d'accès de l'utilisateur IAM en cas de besoin, par exemple lorsqu'un employé quitte votre entreprise. Les utilisateurs IAM peuvent mettre à jour leurs propres clés d'accès s'ils ont obtenu les autorisations nécessaires.

Pour plus d'informations concernant l'octroi aux utilisateurs IAM d'autorisations de mise à jour de leurs propres clés d'accès, veuillez consulter AWS: permet aux utilisateurs IAM de gérer leur propre mot de passe, leurs clés d'accès et leurs clés publiques SSH sur la page Informations d'identification de sécurité. Vous pouvez également appliquer une politique de mots de passe à votre compte pour exiger que tous vos utilisateurs IAM mettent périodiquement à jour leurs mots de passe et la fréquence à laquelle ils doivent le faire. Pour plus d'informations, consultez Définition d'une politique de mot de passe du compte pour les utilisateurs IAM.

Rubriques

- Mise à jour des clés d'accès pour un utilisateur IAM (console)
- Mise à jour des clés d'accès (AWS CLI)
- Mise à jour des clés d'accès (AWS API)

Mise à jour des clés d'accès pour un utilisateur IAM (console)

Vous pouvez mettre à jour les clés d'accès à partir de l' AWS Management Console.

Pour mettre à jour les clés d'accès pour un utilisateur IAM sans interrompre vos applications (console)

1. Pendant que la première clé d'accès est encore active, créez une seconde clé d'accès.
    a. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
    b. Dans le panneau de navigation, choisissez utilisateurs.
    c. Choisissez le nom utilisateur concerné, puis sélectionnez l'onglet Informations d'identification de sécurité.
    d. Dans la section Clés d'accès, choisissez Créer une clé d'accès. Sur la page des bonnes pratiques et alternatives en matière de clés d'accès, choisissez Other (Autre), puis Next (Suivant).
    e. (Facultatif) Définissez une valeur de balise de description pour la clé d'accès afin d'ajouter une paire clé-valeur de balise à cet utilisateur IAM. Cela peut vous aider à identifier et à mettre à jour les clés d'accès par la suite. La clé de balise est définie sur l'identifiant de la clé d'accès. La valeur de balise est définie selon la description de la clé d'accès que vous spécifiez. Lorsque vous avez terminé, sélectionnez Create access key (Créer la clé d'accès).

f. Sur la page Retrieve access keys (Récupérer les clés d'accès), choisissez Show (Afficher) (pour révéler la valeur de la clé d'accès secrète de votre utilisateur) ou Download .csv file (Télécharger le fichier .csv). C'est le seul moment où vous pouvez enregistrer votre clé d'accès secrète. Après avoir enregistré votre clé d'accès secrète dans un emplacement sécurisé, sélectionnez Done (Terminé).

   Lorsque vous créez une clé d'accès pour votre utilisateur, cette paire de clés est activée par défaut et votre utilisateur peut immédiatement l'utiliser. À ce stade, l'utilisateur dispose de deux clés d'accès actives.

2. Mettez à jour toutes les applications et tous les outils pour utiliser la nouvelle clé d'accès.
3. Déterminez si la première clé d'accès est toujours utilisée en consultant les informations Dernière utilisation de la clé d'accès la plus ancienne. Une des approches possibles consiste à attendre plusieurs jours, puis à vérifier si l'ancienne clé d'accès a été utilisée avant de poursuivre.
4. Même si la valeur des informations Dernière utilisation indique que l'ancienne clé n'a jamais été utilisée, nous vous recommandons de ne pas supprimer immédiatement la première clé d'accès. À la place, sélectionnez Actions, puis Deactivate (Désactiver) pour désactiver la première clé d'accès.
5. Utilisez uniquement la nouvelle clé d'accès pour confirmer que vos applications fonctionnent. Toutes les applications et tous les outils qui utilisent encore la clé d'accès d'origine cesseront de fonctionner à ce stade car ils n'ont plus accès aux AWS ressources. Si vous rencontrez l'une de ces applications ou l'un de ces outils, vous pouvez réactiver la première clé d'accès. Revenez ensuite à Step 3 et mettez à jour cette application afin d'utiliser la nouvelle clé.
6. Après avoir attendu un certain temps pour vous assurer que toutes les applications et tous les outils ont été mis à jour, vus pouvez supprimer la première clé d'accès:

   a. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
   b. Dans le panneau de navigation, choisissez utilisateurs.
   c. Choisissez le nom utilisateur concerné, puis sélectionnez l'onglet Informations d'identification de sécurité.
   d. Dans la section Access keys (Clés d'accès) correspondant à la clé d'accès que vous souhaitez supprimer, choisissez Actions, puis Delete (Supprimer). Suivez les instructions de la boîte de dialogue pour tout d'abord désactiver, puis confirmer la suppression.

Pour déterminer quelles clés d'accès doivent être mises à jour ou supprimées (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Au besoin, ajoutez la colonne Access key age (Âge de la clé d'accès) à la table des utilisateurs en procédant comme suit :

   a. Au-dessus de la table à l'extrême droite, choisissez l'icône des paramètres ( ).

   b. Dans Manage columns (Gérer les colonnes), sélectionnez Access key age (Âge de la clé d'accès).

   c. Choisissez Fermer pour revenir à la liste des utilisateurs.
4. La colonne Access key age (Âge de la clé d'accès) affiche le nombre de jours écoulés depuis la création de la clé d'accès active la plus ancienne. Vous pouvez utiliser ces informations pour trouver les utilisateurs dont les clés d'accès doivent être mises à jour ou supprimées. La colonne affiche Aucun pour les utilisateurs sans clé d'accès.

Mise à jour des clés d'accès (AWS CLI)

Vous pouvez mettre à jour les clés d'accès à partir de l' AWS Command Line Interface.

Pour mettre à jour les clés d'accès sans interrompre vos applications (AWS CLI)

1. Pendant que la première clé d'accès est encore active, créez une seconde clé d'accès qui est active par défaut. Exécutez la commande suivante :

   - `aws iam create-access-key`

   À ce stade, l'utilisateur dispose de deux clés d'accès actives.
2. Mettez à jour toutes les applications et tous les outils pour utiliser la nouvelle clé d'accès.
3. Déterminez si la première clé d'accès est toujours utilisée à l'aide de cette commande :

   - `aws iam get-access-key-last-used`

   Une des approches possibles consiste à attendre plusieurs jours, puis à vérifier si l'ancienne clé d'accès a été utilisée avant de poursuivre.

4. Même si l'étape Step 3 n'indique aucune utilisation de l'ancienne clé, nous vous recommandons de ne pas supprimer immédiatement la première clé d'accès. Définissez plutôt l'état de la première clé d'accès sur `Inactive` à l'aide de cette commande :

   - `aws iam update-access-key`

5. Utilisez uniquement la nouvelle clé d'accès pour confirmer que vos applications fonctionnent. Toutes les applications et tous les outils qui utilisent encore la clé d'accès d'origine cesseront de fonctionner à ce stade car ils n'ont plus accès aux AWS ressources. Si vous rencontrez l'une de ces applications ou l'un de ces outils, vous pouvez définir à nouveau son état sur `Active` pour réactiver la première clé d'accès. Revenez ensuite à l'étape Step 2 et mettez à jour cette application afin d'utiliser la nouvelle clé.

6. Après avoir attendu un certain temps pour vous assurer que toutes les applications et tous les outils ont été mis à jour, vus pouvez supprimer la première clé d'accès à l'aide de cette commande :

   - `aws iam delete-access-key`

Mise à jour des clés d'accès (AWS API)

Vous pouvez mettre à jour les clés d'accès à l'aide de l' AWS API.

Pour mettre à jour les clés d'accès sans interrompre vos applications (AWS API)

1. Pendant que la première clé d'accès est encore active, créez une seconde clé d'accès qui est active par défaut. Appelez l'opération suivante :

   - `CreateAccessKey`

   À ce stade, l'utilisateur dispose de deux clés d'accès actives.

2. Mettez à jour toutes les applications et tous les outils pour utiliser la nouvelle clé d'accès.

3. Déterminez si la première clé d'accès est toujours utilisée en appelant cette opération :

   - `GetAccessKeyLastUsed`

   Une des approches possibles consiste à attendre plusieurs jours, puis à vérifier si l'ancienne clé d'accès a été utilisée avant de poursuivre.

4. Même si l'étape Step 3 n'indique aucune utilisation de l'ancienne clé, nous vous recommandons de ne pas supprimer immédiatement la première clé d'accès. Définissez plutôt l'état de la première clé d'accès sur `Inactive` en appelant cette opération :

   - `UpdateAccessKey`
5. Utilisez uniquement la nouvelle clé d'accès pour confirmer que vos applications fonctionnent. Toutes les applications et tous les outils qui utilisent encore la clé d'accès d'origine cesseront de fonctionner à ce stade car ils n'ont plus accès aux AWS ressources. Si vous rencontrez l'une de ces applications ou l'un de ces outils, vous pouvez définir à nouveau son état sur `Active` pour réactiver la première clé d'accès. Revenez ensuite à l'étape Step 2 et mettez à jour cette application afin d'utiliser la nouvelle clé.
6. Après avoir attendu un certain temps pour vous assurer que toutes les applications et tous les outils ont été mis à jour, vus pouvez supprimer la première clé d'accès en appelant cette opération :

   - `DeleteAccessKey`

Sécurisation des clés d'accès

Toute personne en possession de vos clés d'accès a le même niveau d'accès à vos AWS ressources que vous. Par conséquent, AWS elle met tout en œuvre pour protéger vos clés d'accès et, conformément à notre modèle de responsabilité partagée, vous devriez faire de même.

Développez les sections suivantes pour obtenir des conseils qui vous aideront à protéger vos clés d'accès.

> **Note**
>
> Votre organisation peut avoir des exigences de sécurité et des stratégies différentes de celles décrites dans cette rubrique. Les suggestions fournies ici doivent être considérées comme des directives générales.

Supprimer (ou ne pas générer) les clés Utilisateur racine d'un compte AWS d'accès

Le meilleur moyen de protéger votre compte est de n'avoir aucune clé d'accès pour votre Utilisateur racine d'un compte AWS. À moins d'avoir impérativement besoin des clés d'accès de l'utilisateur root (ce qui est rare), il est préférable de ne pas les générer. Créez plutôt un utilisateur administratif pour

les tâches administratives quotidiennes. AWS IAM Identity Center Pour plus d'informations sur la création d'un utilisateur administratif dans IAM Identity Center, voir Getting started dans le guide de l'utilisateur d'IAM Identity Center.

Si vous disposez déjà de clés d'accès d'utilisateur root pour votre compte, nous vous recommandons ce qui suit : recherchez dans vos applications les endroits où vous utilisez actuellement des clés d'accès (le cas échéant) et remplacez les clés d'accès de l'utilisateur root par celles de l'utilisateur IAM. Ensuite, désactivez et supprimez les clés d'accès de l'utilisateur root. Pour plus d'informations sur les modalités de mise à jour des clés d'accès, veuillez consulter Mise à jour des clés d'accès.

Utiliser les informations d'identification de sécurité temporaires (rôles IAM) au lieu des clés d'accès à long terme

Dans de nombreux cas, vous n'avez pas besoin d'une clé d'accès à long terme qui n'expire jamais (comme dans le cas d'un utilisateur IAM). Au lieu de cela, vous pouvez créer des rôles IAM et générer des informations d'identification de sécurité temporaires. Les informations d'identification de sécurité temporaires consistent en un ID de clé d'accès et une clé d'accès secrète, mais elles comprennent également un jeton de sécurité qui indique la date d'expiration des informations d'identification.

Les clés d'accès à long terme, telles que celles associées aux utilisateurs IAM et à l'utilisateur root, demeurent valides jusqu'à ce que vous les révoquiez manuellement. Toutefois, les informations d'identification de sécurité temporaires obtenues via les rôles IAM et d'autres fonctionnalités de l'IAM AWS Security Token Service expirent après un court laps de temps. Utilisez les informations d'identification de sécurité temporaires pour vous aider à réduire les risques en cas de compromission accidentelle des informations d'identification.

Utilisez un rôle IAM et les informations d'identification de sécurité temporaires dans les cas suivants :

- Vous avez une application ou AWS CLI des scripts exécutés sur une instance Amazon EC2. N'utilisez pas les clés d'accès directement dans votre application. Ne transmettez pas les clés d'accès à l'application, ne les intégrez pas à l'application et ne laissez pas l'application les lire depuis n'importe quelle source. Définissez plutôt un rôle IAM qui dispose des autorisations adéquates pour votre application et lancez l'instance Amazon Elastic Compute Cloud (Amazon EC2) avec des rôles pour EC2. Cela permet d'associer un rôle IAM à l'instance Amazon EC2. Cette pratique permet également à l'application d'obtenir des informations d'identification de sécurité temporaires qu'elle peut à son tour utiliser pour effectuer des appels à AWS. Les AWS SDK et AWS Command Line Interface (AWS CLI) peuvent obtenir automatiquement des informations d'identification temporaires à partir du rôle.

- Vous devez accorder l'accès entre comptes. Utilisez un rôle IAM pour établir une relation d'approbation entre les comptes, puis accordez aux utilisateurs d'un compte des autorisations limitées pour accéder au compte approuvé. Pour plus d'informations, consultez Didacticiel IAM : déléguer l'accès entre des comptes AWS à l'aide des rôles IAM.
- Vous disposez d'une application mobile. N'intégrez pas les clés d'accès à l'application, même dans un espace de stockage chiffré. Au lieu de cela, utilisez Amazon Cognito pour gérer les identités de l'utilisateur dans votre application. Ce service vous permet d'authentifier les utilisateurs à l'aide de Login with Amazon, Facebook, Google ou tout autre fournisseur d'identité compatible avec OpenID Connect (OIDC). Vous pouvez ensuite utiliser le fournisseur d'informations d'identification d'Amazon Cognito pour gérer les informations d'identification que votre application utilise pour effectuer des requêtes à AWS.
- Vous souhaitez vous fédérer dans le protocole SAML AWS 2.0 et votre organisation le prend en charge. Si vous travaillez pour une organisation disposant d'un fournisseur d'identité prenant en charge SAML 2.0, configurez le fournisseur pour qu'il utilise SAML. Vous pouvez utiliser le protocole SAML pour échanger des informations d'authentification avec un ensemble d'informations d'identification de sécurité temporaires AWS et en récupérer un. Pour plus d'informations, consultez Fédération SAML 2.0.
- Vous souhaitez vous fédérer dans un magasin d'identités sur site AWS et votre organisation dispose d'un magasin d'identités. Si les utilisateurs peuvent s'authentifier au sein de votre organisation, vous pouvez créer une application qui peut leur délivrer des informations d'identification de sécurité temporaires pour accéder aux AWS ressources. Pour plus d'informations, consultez Activation de l'accès à la AWS console par un courtier d'identité personnalisé.

> **Note**
>
> Utilisez-vous une instance Amazon EC2 avec une application qui nécessite un accès programmatique aux ressources ? AWS Dans ce cas, utilisez les rôles IAM pour EC2.

Gérer correctement les clés d'accès d'un utilisateur IAM

Si vous devez créer des clés d'accès pour un accès programmatique AWS, créez-les pour les utilisateurs IAM, en leur accordant uniquement les autorisations dont ils ont besoin.

Respectez les précautions suivantes pour protéger les clés d'accès des utilisateurs IAM :

- N'intégrez pas de clés d'accès directement dans le code. Les kits AWS SDK et les outils de ligne de commande AWS vous permettent de placer les clés d'accès à des emplacements connus, afin que vous n'ayez pas à les conserver dans le code.

  Placez les clés d'accès à l'un des emplacements suivants :

  - Le fichier AWS d'informations d'identification. Les AWS SDK utilisent AWS CLI automatiquement les informations d'identification que vous stockez dans le fichier AWS d'informations d'identification.

    Pour plus d'informations sur l'utilisation du fichier AWS d'informations d'identification, consultez la documentation de votre SDK. Les exemples incluent Définir les AWS informations d'identification et la région dans le guide du AWS SDK for Java développeur et les fichiers de configuration et d'identification dans le guide de l'AWS Command Line Interface utilisateur.

    Pour stocker les informations d'identification des AWS SDK for .NET et AWS Tools for Windows PowerShell, nous vous recommandons d'utiliser le SDK Store. Pour plus d'informations, veuillez consulter la rubrique Using the SDK Store dans le Guide du développeur AWS SDK for .NET .
  - Variables d'environnement. Sur un système à locataires multiples, choisissez des variables d'environnement utilisateur, et non pas des variables d'environnement système.

    Pour plus d'informations sur l'utilisation des variables d'environnement pour stocker les informations d'identification, veuillez consulter la rubrique Environment Variables dans le Guide de l'utilisateur AWS Command Line Interface .
- Utilisez des clés d'accès différentes pour chaque application. Procédez ainsi afin de pouvoir isoler les autorisations et révoquer les clés d'accès pour des applications individuelles si elles sont compromises. Le fait d'avoir des clés d'accès distinctes pour différentes applications génère également des entrées distinctes dans les fichiers journaux AWS CloudTrail. Cette configuration vous permet d'identifier plus facilement quelle application a effectué des actions spécifiques.
- Mettez à jour les clés d'accès en cas de besoin. Si la clé d'accès risque d'être compromise, mettez-la à jour et supprimez la clé d'accès précédente. Pour plus d'informations, consultez Mise à jour des clés d'accès.
- Supprimez les clés d'accès inutilisées. Si un utilisateur quitte votre organisation, supprimez l'utilisateur IAM correspondant afin qu'il ne puisse plus accéder à vos ressources. Pour savoir quand une clé d'accès a été utilisée pour la dernière fois, utilisez l'`GetAccessKeyLastUsed`API (AWS CLI command : `aws iam get-access-key-last-used`).
- Utilisez des informations d'identification temporaires et configurez l'authentification multifactorielle pour vos opérations d'API les plus sensibles. Avec les politiques IAM, vous pouvez spécifier

quelles opérations d'API un utilisateur est autorisé à appeler. Dans certains cas, vous souhaiterez peut-être bénéficier d'une sécurité supplémentaire en exigeant que les utilisateurs soient authentifiés par le biais de la AWS MFA avant de les autoriser à effectuer des actions particulièrement sensibles. Par exemple, vous disposez peut-être d'une politique qui autorise l'utilisateur à exécuter les actions Amazon EC2 `RunInstances`, `DescribeInstances` et `StopInstances`. Mais vous souhaiterez peut-être restreindre une action destructrice telle que `TerminateInstances` et vous assurer que les utilisateurs ne peuvent effectuer cette action que s'ils s'authentifient auprès d'un périphérique AWS MFA. Pour plus d'informations, consultez Configuration de l'accès aux API protégé par MFA.

Accédez à l'application mobile à l'aide des touches AWS d'accès

Vous pouvez accéder à un ensemble limité de AWS services et de fonctionnalités à l'aide de l'application AWS mobile. L'application mobile vous aide à prendre en charge la réponse aux incidents pendant vos déplacements. Pour de plus amples informations et pour télécharger l'application, veuillez consulter Console AWS pour les applications mobiles.

Vous pouvez vous connecter à l'application mobile à l'aide du mot de passe de votre console ou de vos clés d'accès. En guise de bonne pratique, n'utilisez pas les clés d'accès de l'utilisateur root. Nous vous recommandons vivement, en plus d'utiliser un mot de passe ou un verrou biométrique sur votre appareil mobile, de créer un utilisateur IAM spécifiquement chargé de gérer les AWS ressources à l'aide de l'application mobile. Si vous perdez votre appareil mobile, vous pouvez supprimer l'accès de l'utilisateur IAM.

Pour vous connecter à l'aide des clés d'accès (application mobile)

1. Ouvrez l'application sur votre appareil mobile.
2. Si c'est la première fois que vous ajoutez une identité à l'appareil, choisissez Ajouter une identité, puis cliquez sur Clés d'accès.

   Si vous vous êtes déjà connecté à l'aide d'une autre identité, choisissez l'icône de menu et choisissez Changer d'identité. Choisissez ensuite Se connecter en tant qu'identité différente, puis Clés d'accès.
3. Sur la page Clés d'accès, saisissez vos informations :

   - ID de clé d'accès : saisissez votre ID de clé d'accès.
   - Clé d'accès secrète : saisissez votre clé d'accès secrète.

- Nom de l'identité : saisissez le nom de l'identité qui apparaîtra dans l'application mobile. Elle ne doit pas nécessairement correspondre à votre nom d'utilisateur IAM.
- Code PIN d'identité : créez un numéro d'identification personnel (PIN) que vous utiliserez pour les prochaines connexions.

> **Note**
>
> Si vous activez la biométrie pour l'application AWS mobile, vous serez invité à utiliser votre empreinte digitale ou votre reconnaissance faciale pour la vérification au lieu du code PIN. Si la biométrie échoue, vous pouvez être invité à entrer le code PIN à la place.

4. Choisissez Vérifier et ajouter des clés.

   Vous pouvez désormais accéder à un ensemble sélectionné de vos ressources à l'aide de l'application mobile.

Informations connexes

Les rubriques suivantes fournissent des conseils pour configurer les AWS SDK et pour utiliser les clés AWS CLI d'accès :

- Définissez AWS les informations d'identification et la région dans le guide du AWS SDK for Java développeur
- Using the SDK Store dans le Guide du développeur AWS SDK for .NET
- Providing Credentials to the SDK dans le Guide du développeur AWS SDK for PHP
- Configuration dans la documentation de Boto 3 (AWS SDK pour Python)
- Using AWS Credentials dans le Guide de l'utilisateur AWS Tools for Windows PowerShell
- Configuration and credential files dans le Guide de l'utilisateur AWS Command Line Interface
- Granting access using an IAM role dans le Guide du développeur AWS SDK for .NET
- Configure IAM roles for Amazon EC2 dans le AWS SDK for Java 2.x

Audit des clés d'accès

Vous pouvez vérifier les clés AWS d'accès contenues dans votre code pour déterminer si elles proviennent d'un compte que vous possédez. Vous pouvez transmettre un identifiant de clé

d'accès à l'aide de la `aws sts get-access-key-info` AWS CLI commande ou de l'opération `GetAccessKeyInfo` AWS API.

Les opérations AWS CLI et AWS API renvoient l'ID du Compte AWS auquel appartient la clé d'accès. Les ID de clé d'accès commençant par `AKIA` sont les informations d'identification à long terme d'un utilisateur IAM ou d'un Utilisateur racine d'un compte AWS. Les identifiants de clé d'accès commençant par `ASIA` sont des informations d'identification temporaires créées à l'aide d' AWS STS opérations. Si le compte de la réponse vous appartient, vous pouvez vous connecter en tant qu'utilisateur racine et vérifier vos clés d'accès d'utilisateur racine. Ensuite, vous pouvez extraire un rapport d'informations d'identification pour savoir quel utilisateur IAM possède les clés. Pour savoir qui a demandé les informations d'identification temporaires pour une clé d'`ASIA`accès, consultez les AWS STS événements dans vos CloudTrail journaux.

Pour des raisons de sécurité, vous pouvez consulter AWS CloudTrail les journaux pour savoir qui a effectué une action dans AWS. Vous pouvez utiliser la clé de condition `sts:SourceIdentity` dans la politique d'approbation de rôle pour exiger des utilisateurs qu'ils spécifient une identité lorsqu'ils endossent un rôle. Par exemple, vous pouvez exiger que les utilisateurs IAM spécifient leur propre nom d'utilisateur comme identité de source. Cela peut vous aider à déterminer quel utilisateur a effectué une action spécifique dans AWS. Pour plus d'informations, consultez `sts:SourceIdentity`.

Cette opération n'indique pas l'état de la clé d'accès. La clé peut être active, inactive ou supprimée. Les clés actives peuvent ne pas avoir les autorisations nécessaires pour effectuer une opération. La fourniture d'une clé d'accès supprimée peut renvoyer une erreur indiquant que la clé n'existe pas.

Réinitialisation de mots de passe ou de clés d'accès perdus ou oubliés pour AWS

> **Important**
>
> Vous rencontrez des difficultés pour vous connecter à AWS ? Assurez-vous que vous êtes sur la bonne page de connexion AWS pour votre type d'utilisateur. Si vous êtes le Utilisateur racine d'un compte AWS (propriétaire du compte), vous pouvez vous connecter à AWS l'aide des informations d'identification que vous avez définies lors de la création du Compte AWS. Si vous êtes un utilisateur IAM, votre administrateur de compte peut vous fournir les informations d'identification que vous pouvez utiliser pour vous connecter à AWS. Si vous avez besoin d'assistance, n'utilisez pas le lien de commentaires sur cette page, car le formulaire est reçu par l'équipe de AWS documentation AWS Support. À la place, sur la

page Contactez-nous, choisissez Still unable to log in your account AWS (Impossible de se connecter à votre compte), puis choisissez l'une des options d'assistance disponibles.

Sur la page de connexion principale, vous devez saisir votre adresse électronique pour vous connecter en tant qu'utilisateur racine, ou votre ID de compte pour vous connecter en tant qu'utilisateur IAM. Vous pouvez uniquement fournir votre mot de passe sur la page de connexion correspondant à votre type d'utilisateur. Pour plus d'informations, consultez Connexion à la AWS Management Console .

Si vous êtes sur la page de connexion appropriée et que vous perdez ou oubliez vos mots de passe ou vos clés d'accès, vous ne pouvez pas les récupérer depuis IAM. Par contre, vous pouvez les réinitialiser à l'aide des méthodes suivantes :

- Utilisateur racine d'un compte AWS mot de passe — Si vous oubliez le mot de passe de votre utilisateur root, vous pouvez le réinitialiser depuis le AWS Management Console. Pour plus de détails, consultez the section called “Réinitialisation d'un mot de passe d'utilisateur racine perdu ou oublié” plus loin dans cette rubrique.
- Compte AWS clés d'accès — Si vous oubliez les clés d'accès à votre compte, vous pouvez créer de nouvelles clés d'accès sans désactiver les clés d'accès existantes. Si vous n'utilisez pas les clés existantes, vous pouvez les supprimer. Pour plus d'informations, consultez Création de clés d'accès pour l'utilisateur racine et Suppression des clés d'accès pour l'utilisateur racine.
- IAM user password (Mot de passe d'utilisateur) : si vous êtes un utilisateur IAM et que vous oubliez votre mot de passe, vous devez demander à votre administrateur de le réinitialiser. Pour en savoir comment un administrateur peut gérer votre mot de passe, consultez Gestion des mots de passe des utilisateurs IAM.
- IAM user access keys (Clés d'accès d'utilisateur) : si vous êtes un utilisateur IAM et que vous oubliez vos clés d'accès, vous aurez besoin de nouvelles clés d'accès. Si vous êtes autorisé à créer vos propres clés d'accès, vous trouverez les instructions permettant d'en créer de nouvelles dans Gestion des clés d'accès (console). Si vous ne disposez pas des autorisations requises, vous devez demander à votre administrateur de créer de nouvelles clés d'accès. Si vous utilisez encore vos anciennes clés, demandez à votre administrateur de ne pas les supprimer. Pour en savoir comment un administrateur peut gérer vos clés d'accès, consultez Gestion des clés d'accès pour les utilisateurs IAM.

Utilisation de l'authentification multifactorielle (MFA) dans AWS

Follow us on Twitter

Pour une sécurité accrue, nous vous recommandons de configurer l'authentification multifactorielle (MFA) afin de protéger AWS vos ressources. Vous pouvez activer le MFA pour les utilisateurs Utilisateur racine d'un compte AWS et IAM. Lorsque vous activez l'authentification MFA pour l'utilisateur root, cela affecte uniquement les informations d'identification de l'utilisateur root. Les utilisateurs IAM du compte sont des identités distinctes avec leurs propres informations d'identification, et chaque identité dispose de sa propre configuration MFA.

Vous pouvez enregistrer jusqu'à huit dispositifs MFA de n'importe quelle combinaison des types MFA actuellement pris en charge avec votre Utilisateur racine d'un compte AWS et les utilisateurs IAM. Pour obtenir plus d'informations sur le types MFA pris en charge, consultez Types de MFA disponibles pour les utilisateurs d'IAM. Avec plusieurs appareils MFA, un seul appareil MFA est nécessaire pour se connecter AWS Management Console ou créer une session via cet utilisateur. AWS CLI

> **Note**
>
> Nous vous recommandons de demander à vos utilisateurs humains d'utiliser des informations d'identification temporaires lors de l'accès AWS. Avez-vous envisagé d'en utiliser AWS IAM Identity Center ? Vous pouvez utiliser IAM Identity Center pour gérer de manière centralisée l'accès à plusieurs comptes Comptes AWS et fournir aux utilisateurs un accès par authentification unique protégé par le MFA à tous les comptes qui leur sont attribués à partir d'un seul endroit. Avec IAM Identity Center, vous pouvez créer et gérer les identités des utilisateurs dans IAM Identity Center ou vous connecter facilement à votre fournisseur d'identité compatible SAML 2.0 existant. Pour plus d'informations, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Types de MFA disponibles pour les utilisateurs d'IAM

La MFA renforce la sécurité car elle oblige les utilisateurs à fournir une authentification unique à partir d'un mécanisme AWS MFA compatible en plus de leurs informations de connexion habituelles lorsqu'ils accèdent à des sites Web ou à des services. AWS AWS prend en charge les types MFA suivants : clés d'accès et clés de sécurité, applications d'authentification virtuelle et jetons TOTP matériels.

Clés d'accès et clés de sécurité

AWS Identity and Access Management prend en charge les clés d'accès et les clés de sécurité pour la MFA. Basées sur les normes FIDO, les clés d'accès utilisent la cryptographie à clé publique pour fournir une authentification solide, résistante au hameçonnage et plus sécurisée que les mots de passe. AWS prend en charge deux types de clés d'accès : les clés d'accès liées à l'appareil (clés de sécurité) et les clés d'accès synchronisées.

- Clés de sécurité : il s'agit de périphériques physiques YubiKey, tels que a, utilisés comme deuxième facteur d'authentification.
- Clés d'accès synchronisées : elles utilisent des gestionnaires d'identifiants provenant de fournisseurs tels que Google, Apple, Microsoft et de services tiers tels que 1Password, Dashlane et Bitwarden comme deuxième facteur.

Vous pouvez utiliser des authentificateurs biométriques intégrés, tels que Touch ID sur Apple MacBooks et la reconnaissance faciale Windows Hello sur PC, pour déverrouiller votre gestionnaire d'identifiants et vous y connecter. AWS Les clés d'accès sont créées avec le fournisseur de votre choix à l'aide de votre empreinte digitale, de votre visage ou du code PIN de votre appareil. Vous pouvez synchroniser les clés d'accès sur tous vos appareils pour faciliter les connexions et améliorer la convivialité AWS et la récupérabilité.

La FIDO Alliance tient à jour une liste de tous les produits certifiés FIDO qui sont compatibles avec les spécifications FIDO. Une clé d'accès ou une clé de sécurité unique prend en charge plusieurs comptes utilisateur root et utilisateurs IAM. Pour plus d'informations sur l'activation des clés d'accès et des clés de sécurité pour un utilisateur IAM, consultez. Activation d'une clé d'accès ou d'une clé de sécurité (console)

Applications d'authentification virtuelle

Une application d'authentification virtuelle s'exécute sur un téléphone ou un autre appareil et émule un appareil physique. Les applications d'authentification virtuelle mettent en œuvre l'algorithme TOTP (mot de passe unique à durée limitée) et prennent en charge plusieurs jetons sur un seul dispositif. L'utilisateur doit saisir un code valide sur l'appareil lorsqu'il y est invité lors de la connexion. Chaque jeton attribué à un utilisateur doit être unique. Un utilisateur ne peut pas saisir de code à partir du jeton d'un autre utilisateur pour s'authentifier.

Nous vous recommandons d'utiliser un dispositif MFA virtuel pendant l'attente de l'approbation d'achat du matériel ou pendant que vous attendez de recevoir votre matériel. Pour obtenir la

liste de quelques applications prises en charge que vous pouvez utiliser comme appareils MFA virtuels, consultez la section Multi-Factor Authentication (MFA). Pour obtenir des instructions sur la configuration d'un périphérique MFA virtuel pour un utilisateur IAM, consultez. Activation d'un appareil Multi-Factor Authentication (MFA) virtuel (console)

Tokens TOTP matériels

Un périphérique matériel génère un code numérique à six chiffres basé sur l'algorithme TOTP (mot de passe à usage unique basé sur le temps). L'utilisateur doit saisir un code valide à partir du dispositif sur une deuxième page web lors de la connexion. Chaque dispositif MFA attribué à un utilisateur doit être unique. Un utilisateur ne peut pas saisir un code à partir du périphérique d'un autre utilisateur pour s'authentifier. Pour plus d'informations sur les dispositifs matériels MFA pris en charge, consultez la section Multi-Factor Authentication (MFA). Pour obtenir des instructions sur la configuration d'un jeton TOTP matériel pour un utilisateur IAM, consultez. Activation d'un jeton TOTP matériel (console)

Si vous souhaitez utiliser un périphérique MFA physique, nous vous recommandons d'utiliser des clés de sécurité comme alternative aux périphériques TOTP matériels. Les clés de sécurité offrent l'avantage de ne pas nécessiter de batterie, de résister au phishing et de prendre en charge plusieurs utilisateurs root et IAM sur un seul appareil pour une sécurité renforcée.

> **Note**
>
> MFA basé sur les SMS : AWS a cessé de prendre en charge l'authentification multifactorielle (MFA) par SMS. Nous recommandons aux clients dont les utilisateurs IAM utilisent l'authentification MFA basée sur des SMS de passer à l'une des méthodes alternatives suivantes : clé d'accès ou clé de sécurité, périphérique MFA virtuel (logiciel) ou périphérique MFA matériel. Vous pouvez identifier les utilisateurs dans votre compte avec un dispositif MFA SMS affecté. Pour ce faire, accédez à la console IAM, choisissez Utilisateurs dans le panneau de navigation, puis recherchez les utilisateurs avec SMS mentionné dans la colonne MFA de la table.

Rubriques

- Activation des appareils MFA pour les utilisateurs de AWS
- Vérification du statut de l'authentification MFA
- Resynchronisation de dispositifs MFA virtuels et matériels

- Désactivation des dispositifs MFA
- Que faire si un dispositif MFA est perdu ou cesse de fonctionner ?
- Configuration de l'accès aux API protégé par MFA
- Exemple de code : demande d'informations d'identification avec l'authentification multifacteur

Activation des appareils MFA pour les utilisateurs de AWS

Les étapes de configuration de dispositifs MFA dépendent du type de dispositif MFA que vous utilisez.

Rubriques

- Étapes générales de l'activation de dispositifs MFA
- Activation d'une clé d'accès ou d'une clé de sécurité (console)
- Activation d'un appareil Multi-Factor Authentication (MFA) virtuel (console)
- Activation d'un jeton TOTP matériel (console)
- Activation et gestion des appareils MFA virtuels (AWS CLI ou AWS API)

Étapes générales de l'activation de dispositifs MFA

La procédure générale suivante décrit comment configurer et utiliser l'authentification MFA et fournit des liens vers des informations connexes.

> **Remarque**
>
> Vous pouvez également regarder cette vidéo en anglais intitulée How to Setup AWS Multi-Factor Authentication (MFA) AWS and Budget Alerts, pour plus d'informations.

1. Obtenir un dispositif MFA tels que l'un des suivants. Vous pouvez activer jusqu'à huit appareils MFA par utilisateur Utilisateur racine d'un compte AWS ou par utilisateur IAM de n'importe quelle combinaison des types suivants.
   - Un dispositif MFA virtuel, qui est une application logicielle conforme à RFC 6238, un algorithme compatible avec la norme TOTP (Time-Based One-Time Password). Vous pouvez installer l'application sur un téléphone ou un autre appareil. Pour obtenir une liste des applications que vous pouvez utiliser comme dispositifs MFA virtuels, consultez Multi-Factor Authentication.

- Une clé d'accès ou une clé de sécurité dont la configuration est AWS prise en charge. La FIDO Alliance tient à jour une liste de tous les produits certifiés FIDO qui sont compatibles avec les spécifications FIDO.
- Un dispositif MFA basé sur le matériel provenant d'un fournisseur tiers, tel qu'un dispositif à jeton. Ces jetons sont utilisés exclusivement avec Comptes AWS. Pour plus d'informations, consultez Activation d'un jeton TOTP matériel (console). Vous ne pouvez utiliser que des jetons dont les graines uniques sont partagées en toute sécurité avec AWS. Les graines de jetons sont des clés secrètes générées au moment de la production des jetons. Les jetons achetés auprès d'autres sources ne fonctionneront pas avec IAM. Pour garantir la compatibilité, vous devez acheter votre périphérique MFA matériel via l'un des liens suivants : jeton OTP ou carte d'affichage OTP.

2. Activer le dispositif MFA.
   - Jetons TOTP virtuels ou matériels : vous pouvez utiliser des AWS CLI commandes ou des opérations d' AWS API pour activer un dispositif MFA virtuel pour un utilisateur IAM. Vous ne pouvez pas activer de périphérique MFA Utilisateur racine d'un compte AWS avec l' AWS API AWS CLI, Tools for Windows PowerShell ou tout autre outil de ligne de commande. Toutefois, vous pouvez utiliser le AWS Management Console pour activer un périphérique MFA pour l'utilisateur root.
   - Clés de passe et clés de sécurité : les utilisateurs root et les utilisateurs IAM dotés de clés d'accès ou de clés de sécurité peuvent activer AWS Management Console uniquement l' AWS CLI API ou. AWS

   Pour plus d'informations sur l'activation de chaque type de dispositif MFA, consultez les pages suivantes :
   - Dispositif MFA virtuel : Activation d'un appareil Multi-Factor Authentication (MFA) virtuel (console)
   - Clés d'accès et clés de sécurité : Activation d'une clé d'accès ou d'une clé de sécurité (console)
   - Jeton TOTP matériel :Activation d'un jeton TOTP matériel (console)
3. Activer plusieurs dispositifs MFA (recommandé)
   - Nous vous recommandons d'activer plusieurs appareils MFA pour les utilisateurs Utilisateur racine d'un compte AWS et IAM de votre. Comptes AWS Cela vous permet d'élever la sécurité dans votre Comptes AWS et de simplifier la gestion de l'accès aux utilisateurs hautement privilégiés, tels que le Utilisateur racine d'un compte AWS.
   - Vous pouvez enregistrer jusqu'à huit appareils MFA de n'importe quelle combinaison des types de MFA actuellement pris en charge auprès de vos Utilisateur racine d'un compte AWS

utilisateurs et de ceux d'IAM. Avec plusieurs appareils MFA, vous n'avez besoin que d'un seul appareil MFA pour vous connecter AWS Management Console ou créer une session via cet utilisateur. AWS CLI Un utilisateur IAM doit s'authentifier avec un dispositif MFA existant pour activer ou désactiver un dispositif MFA supplémentaire.

- En cas de perte, de vol ou d'inaccessibilité d'un appareil MFA, vous pouvez utiliser l'un des appareils MFA restants pour y accéder Compte AWS sans effectuer la procédure de récupération. Compte AWS En cas de perte ou de vol d'un dispositif MFA, celui-ci doit être dissocié du principal IAM auquel il est associé.
- L'utilisation de plusieurs MFA permet à vos employés travaillant sur des sites géographiquement dispersés ou travaillant à distance d'utiliser l'authentification multifacteur matérielle pour y AWS accéder sans avoir à coordonner l'échange physique d'un seul appareil matériel entre les employés.
- L'utilisation de dispositifs MFA supplémentaires pour les principaux IAM vous permet d'utiliser un ou plusieurs MFA pour un usage quotidien, tout en conservant les dispositifs MFA physiques dans un emplacement physique sécurisé, tel qu'une chambre forte ou un coffre-fort pour la sauvegarde et la redondance.

4. Utilisez le dispositif MFA lorsque vous vous connectez ou accédez aux ressources AWS .

   - Clés d'accès et clés de sécurité — Pour accéder à un AWS site Web, entrez vos informations d'identification, puis, selon le type de clé d'accès dont vous disposez, appuyez sur la clé de sécurité FIDO, entrez le code PIN de l'appareil ou fournissez votre empreinte digitale ou votre visage lorsque vous y êtes invité.
   - Dispositifs MFA virtuels et jetons TOTP matériels — Pour accéder à un AWS site Web, vous avez besoin d'un code MFA provenant de l'appareil en plus de votre nom d'utilisateur et de votre mot de passe.

     Pour accéder aux opérations d'API protégées par un code MFA, vous avez besoin des éléments suivants :

     - Un code MFA
     - L'identifiant du dispositif MFA (numéro de série d'un périphérique physique ou ARN d'un périphérique virtuel défini dans AWS)
     - L'ID de clé d'accès et la clé d'accès secrète habituels

Remarques

- Vous ne pouvez pas transmettre les informations MFA relatives à une clé de sécurité FIDO aux opérations d' AWS STS API pour demander des informations d'identification temporaires.
- Vous ne pouvez pas utiliser de AWS CLI commandes ou AWS d'opérations d'API pour activer les clés de sécurité FIDO.
- Vous ne pouvez pas utiliser le même nom pour plusieurs périphériques racine ou MFA IAM.

Pour plus d'informations, voir Utilisation de dispositifs MFA avec votre page de connexion IAM.

Activation d'une clé d'accès ou d'une clé de sécurité (console)

Les clés d'accès sont un type de dispositif d'authentification multifactorielle (MFA) que vous pouvez utiliser pour protéger vos ressources. AWS AWS prend en charge les clés d'accès synchronisées et les clés d'accès liées à l'appareil, également appelées clés de sécurité.

Les clés d'accès synchronisées permettent aux utilisateurs d'IAM d'accéder à leurs identifiants de connexion FIDO sur bon nombre de leurs appareils, même les plus récents, sans avoir à réinscrire chaque appareil sur chaque compte. Les clés d'accès synchronisées incluent des gestionnaires d'identifiants internes tels que Google, Apple et Microsoft et des gestionnaires d'identifiants tiers tels que 1Password, Dashlane et Bitwarden comme deuxième facteur. Vous pouvez également utiliser la biométrie intégrée à l'appareil (par exemple, TouchID, FaceID, Windows Hello) pour déverrouiller le gestionnaire d'identifiants de votre choix afin d'utiliser des clés d'accès.

Sinon, les clés d'accès liées à l'appareil sont liées à une clé de sécurité FIDO que vous branchez sur un port USB de votre ordinateur, puis que vous tapez lorsque vous y êtes invité pour terminer le processus de connexion en toute sécurité. Si vous utilisez déjà une clé de sécurité FIDO avec d'autres services et que sa configuration est AWS prise en charge (par exemple, la série YubiKey 5 de Yubico), vous pouvez également l'utiliser avec. AWS Sinon, vous devez acheter une clé de sécurité FIDO si vous souhaitez l'utiliser WebAuthn pour l'entrée AWS MFA. De plus, les clés de sécurité FIDO peuvent prendre en charge plusieurs utilisateurs IAM ou root sur le même appareil, améliorant ainsi leur utilité pour la sécurité des comptes. Pour connaître les spécifications et les informations d'achat de ces deux types d'appareils, consultez authentification multifacteur.

Vous pouvez enregistrer jusqu'à huit appareils MFA de n'importe quelle combinaison des types de MFA actuellement pris en charge auprès de vos Utilisateur racine d'un compte AWS utilisateurs et de ceux d'IAM. Avec plusieurs appareils MFA, vous n'avez besoin que d'un seul appareil MFA pour vous connecter AWS Management Console ou créer une session via cet utilisateur. AWS CLI Nous vous recommandons d'enregistrer plusieurs appareils MFA. Par exemple, vous pouvez enregistrer un authentificateur intégré ainsi qu'une clé de sécurité que vous conservez dans un endroit physiquement sûr. Si vous ne pouvez pas à utiliser votre authentificateur intégré, vous pouvez utiliser votre clé de sécurité enregistrée. Pour les applications d'authentification, nous recommandons également d'activer la fonctionnalité de sauvegarde ou de synchronisation dans le cloud dans ces applications afin d'éviter de perdre l'accès à votre compte si vous perdez ou cassez votre appareil avec les applications d'authentification.

> **Note**
>
> Nous vous recommandons de demander à vos utilisateurs humains d'utiliser des informations d'identification temporaires lors de l'accès à AWS. Vos utilisateurs peuvent se fédérer AWS auprès d'un fournisseur d'identité où ils s'authentifient à l'aide de leurs identifiants d'entreprise et de leurs configurations MFA. Pour gérer l'accès aux applications professionnelles AWS et à celles-ci, nous vous recommandons d'utiliser IAM Identity Center. Pour plus d'informations, consultez le guide de l'utilisateur d'IAM Identity Center.

Rubriques

- Autorisations nécessaires
- Activer une clé d'accès ou une clé de sécurité pour votre propre utilisateur IAM (console)
- Activer une clé d'accès ou une clé de sécurité pour un autre utilisateur IAM (console)
- Remplacer une clé d'accès ou une clé de sécurité
- Configurations prises en charge pour l'utilisation des clés d'accès et des clés de sécurité

Autorisations nécessaires

Pour gérer une clé d'accès FIDO pour votre propre utilisateur IAM tout en protégeant les actions sensibles liées au MFA, vous devez disposer des autorisations définies dans la politique suivante :

> **Note**
>
> Les valeurs ARN sont des valeurs statiques et n'indiquent pas le protocole qui a été utilisé pour enregistrer l'authentificateur. Nous avons déconseillé U2F, donc toutes les nouvelles implémentations l'utilisent. WebAuthn

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Activer une clé d'accès ou une clé de sécurité pour votre propre utilisateur IAM (console)

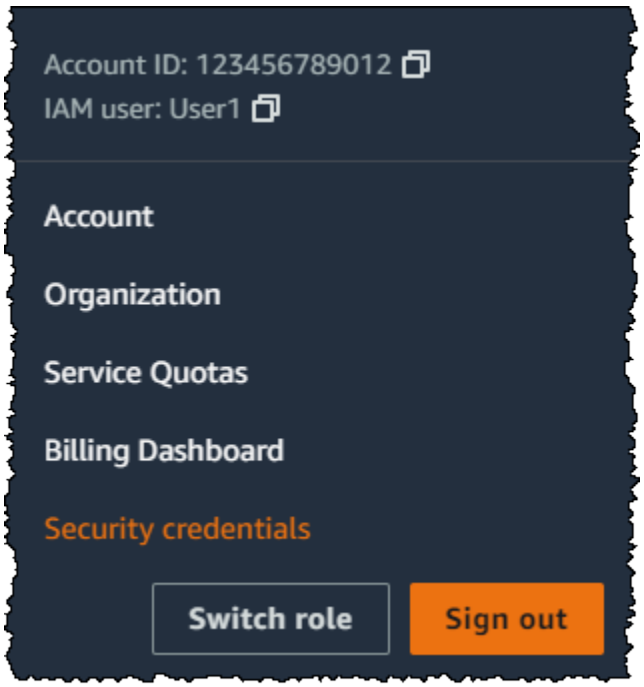
Vous pouvez activer une clé d'accès ou une clé de sécurité pour votre propre utilisateur IAM AWS Management Console uniquement, et non depuis l'API AWS CLI or AWS . Avant de pouvoir activer une clé de sécurité, vous devez avoir un accès physique à l'appareil.

Pour activer une clé d'accès ou une clé de sécurité pour votre propre utilisateur IAM (console)

1. Utilisez votre AWS identifiant ou alias de compte, votre nom d'utilisateur IAM et votre mot de passe pour vous connecter à la console IAM.

   > **Note**
   >
   > Pour votre commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser votre nom d'utilisateur IAM et les informations de votre compte. Si vous vous êtes déjà connecté en tant qu'utilisateur différent, sélectionnez Sign in to a different account (Se connecter à un compte différent) en bas de la page pour revenir à la page de connexion principale. À partir de là, vous pouvez saisir votre identifiant de AWS compte ou votre alias de compte pour être redirigé vers la page de connexion utilisateur IAM de votre compte.

   Pour obtenir votre Compte AWS identifiant, contactez votre administrateur.
2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).



3. Sur la page de l'utilisateur IAM sélectionné, choisissez l'onglet Informations d'identification de sécurité.
4. Dans la section Multi-Factor Authentication (MFA) (Authentification multifactorielle (MFA)), sélectionnez Assign MFA device (Attribuer un dispositif MFA).
5. Sur la page du nom de l'appareil MFA, entrez un nom d'appareil, choisissez Clé d'accès ou Clé de sécurité, puis choisissez Suivant.
6. Dans Configurer l'appareil, configurez votre clé d'accès. Créez une clé d'accès avec des données biométriques telles que votre visage ou votre empreinte digitale, avec le code PIN d'un appareil, ou en insérant la clé de sécurité FIDO dans le port USB de votre ordinateur et en la touchant.
7. Suivez les instructions de votre navigateur, puis choisissez Continuer.

Vous avez maintenant enregistré votre clé d'accès ou votre clé de sécurité pour une utilisation avec AWS. Pour plus d'informations sur l'utilisation de la MFA avec le AWS Management Console, consultez. Utilisation de dispositifs MFA avec votre page de connexion IAM

Activer une clé d'accès ou une clé de sécurité pour un autre utilisateur IAM (console)

Vous pouvez activer une clé d'accès ou une sécurité pour un autre utilisateur IAM AWS Management Console uniquement, et non depuis l'API AWS CLI or AWS .

Pour activer une clé d'accès ou une sécurité pour un autre utilisateur IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Sous Utilisateurs, choisissez le nom de l'utilisateur pour lequel vous souhaitez activer la MFA.
4. Sur la page utilisateur IAM sélectionnée, choisissez l'onglet Security Credentials.
5. Dans la section Multi-Factor Authentication (MFA) (Authentification multifactorielle (MFA)), sélectionnez Assign MFA device (Attribuer un dispositif MFA).
6. Sur la page du nom de l'appareil MFA, entrez un nom d'appareil, choisissez Clé d'accès ou Clé de sécurité, puis choisissez Suivant.
7. Dans Configurer l'appareil, configurez votre clé d'accès. Créez une clé d'accès avec des données biométriques telles que votre visage ou votre empreinte digitale, avec le code PIN d'un appareil, ou en insérant la clé de sécurité FIDO dans le port USB de votre ordinateur et en la touchant.
8. Suivez les instructions de votre navigateur, puis choisissez Continuer.

Vous avez maintenant enregistré un mot de passe ou une clé de sécurité à utiliser par un autre utilisateur IAM. AWS Pour plus d'informations sur l'utilisation de la MFA avec le AWS Management Console, consultez. Utilisation de dispositifs MFA avec votre page de connexion IAM

Remplacer une clé d'accès ou une clé de sécurité

Vous pouvez attribuer à un utilisateur jusqu'à huit dispositifs MFA de n'importe quelle combinaison des types MFA actuellement pris en charge, en même temps que vos Utilisateur racine d'un compte AWS utilisateurs et ceux d'IAM. Si l'utilisateur perd un authentificateur FIDO ou a besoin de le remplacer pour une raison quelconque, vous devez tout d'abord désactiver l'ancien authentificateur FIDO. Vous pouvez alors ajouter un nouveau dispositif MFA pour l'utilisateur.

- Pour désactiver le dispositif actuellement associé à un autre utilisateur IAM, veuillez consulter la rubrique Désactivation des dispositifs MFA.
- Pour ajouter une nouvelle clé de sécurité FIDO pour un utilisateur IAM, veuillez consulter Activer une clé d'accès ou une clé de sécurité pour votre propre utilisateur IAM (console).

Si vous n'avez pas accès à une nouvelle clé d'accès ou à une nouvelle clé de sécurité, vous pouvez activer un nouveau périphérique MFA virtuel ou un nouveau jeton TOTP matériel. Pour plus d'informations, consultez l'un des liens suivants :

- Activation d'un appareil Multi-Factor Authentication (MFA) virtuel (console)
- Activation d'un jeton TOTP matériel (console)

Configurations prises en charge pour l'utilisation des clés d'accès et des clés de sécurité

Vous pouvez utiliser les clés d'accès liées aux appareils FIDO2, également appelées clés de sécurité, comme méthode d'authentification multifactorielle (MFA) avec IAM en utilisant les configurations actuellement prises en charge. Il s'agit notamment des appareils FIDO2 pris en charge par IAM et des navigateurs compatibles FIDO2. Avant d'enregistrer votre appareil FIDO2, vérifiez que vous utilisez la dernière version du navigateur et du système d'exploitation (OS). Les fonctionnalités peuvent se comporter différemment selon les navigateurs, les authentificateurs et les clients du système d'exploitation. Si l'enregistrement de votre appareil échoue sur un navigateur, vous pouvez essayer de vous enregistrer avec un autre navigateur.

FIDO2 est une norme d'authentification ouverte et une extension de FIDO U2F, offrant le même niveau de sécurité élevé basé sur la cryptographie à clé publique. FIDO2 comprend la spécification d'authentification Web (WebAuthn API) du W3C et le protocole CTAP (Client-to-Authenticator Protocol) de FIDO Alliance, un protocole de couche application. CTAP permet la communication entre le client ou la plateforme, comme un navigateur ou un système d'exploitation, avec un authentificateur externe. Lorsque vous activez un authentificateur certifié FIDO AWS, la clé de sécurité crée une nouvelle paire de clés à utiliser uniquement. AWS Tout d'abord, saisissez vos informations d'identification. Lorsque vous y êtes invité, vous appuyez sur la clé de sécurité qui répond au défi d'authentification émis par AWS. Pour en savoir plus sur la norme FIDO2, veuillez consulter la rubrique Projet FIDO2.

Dispositifs FIDO2 pris en charge par AWS

IAM prend en charge les dispositifs de sécurité FIDO2 qui se connectent à vos appareils via USB, Bluetooth ou NFC. IAM prend également en charge les authentificateurs de plateforme tels que TouchID, FaceID ou Windows Hello.

> **Note**
>
> AWS nécessite l'accès au port USB physique de votre ordinateur pour vérifier votre appareil FIDO2. Les clés de sécurité ne fonctionneront pas avec une machine virtuelle, une connexion à distance ou le mode navigation privée d'un navigateur.

La FIDO Alliance tient à jour une liste de tous les produits FIDO2 qui sont compatibles avec les spécifications FIDO.

Navigateurs compatibles avec FIDO2

La disponibilité des dispositifs de sécurité FIDO2 exécutés dans un navigateur Web dépend de la combinaison du navigateur et du système d'exploitation. Les navigateurs suivants prennent actuellement en charge l'utilisation de clés de sécurité :

	macOS 10.15+	Windows 10	Linux	iOS 14.5+	Android 7+
Chrome	Oui	Oui	Oui	Oui	Non
Safari	Oui	Non	Non	Oui	Non
Edge	Oui	Oui	Non	Oui	Non
Firefox	Oui	Oui	Non	Oui	Non

> **Note**
>
> La plupart des versions de Firefox qui supportent actuellement FIDO2 ne l'activent pas par défaut. Pour obtenir des instructions sur l'activation du support FIDO2 dans Firefox, consultez. Résolution des problèmes liés aux clés de sécurité FIDO

Pour plus d'informations sur la prise en charge par navigateur pour un appareil certifié FIDO2 tel que YubiKey, voir Support du système d'exploitation et du navigateur Web pour FIDO2 et U2F.

Plug-ins de navigateur

AWS ne prend en charge que les navigateurs compatibles nativement avec FIDO2. AWS ne prend pas en charge l'utilisation de plugins pour ajouter le support du navigateur FIDO2. Certains plugins de navigateur sont incompatibles avec la norme FIDO2 et peuvent provoquer des résultats inattendus avec les clés de sécurité FIDO2.

Pour plus d'informations sur la désactivation des plug-ins et pour obtenir d'autres conseils de dépannage, consultez Je ne peux pas activer ma clé de sécurité FIDO.

Certifications des appareils

Nous capturons et attribuons les certifications relatives aux appareils, telles que la validation FIPS et le niveau de certification FIDO, uniquement lors de l'enregistrement d'une clé de sécurité. La certification de votre appareil est extraite du FIDO Alliance Metadata Service (MDS). Si le statut ou le niveau de certification de votre clé de sécurité change, cela ne sera pas automatiquement reflété dans les balises de l'appareil. Pour mettre à jour les informations de certification d'un appareil, enregistrez à nouveau l'appareil pour récupérer les informations de certification mises à jour.

AWS fournit les types de certification suivants sous forme de clés de condition lors de l'enregistrement de l'appareil, obtenus à partir de FIDO MDS : niveaux de certification FIPS-140-2, FIPS-140-3 et FIDO. Vous avez la possibilité de spécifier l'enregistrement d'authentificateurs spécifiques dans leurs politiques IAM, en fonction du type et du niveau de certification que vous préférez. Pour plus d'informations, consultez les politiques ci-dessous.

Exemples de politiques pour la certification des appareils

Les cas d'utilisation suivants présentent des exemples de politiques qui vous permettent d'enregistrer des appareils MFA avec des certifications FIPS.

Rubriques

- Cas d'utilisation 1 : autoriser uniquement l'enregistrement des appareils certifiés FIPS-140-2 L2
- Cas d'utilisation 2 : autoriser l'enregistrement des appareils certifiés FIPS-140-2 L2 et FIDO L1
- Cas d'utilisation 3 : autoriser l'enregistrement des appareils certifiés FIPS-140-2 L2 ou FIPS-140-3 L2
- Cas d'utilisation n°4 : autoriser l'enregistrement des appareils certifiés FIPS-140-2 L2 et prenant en charge d'autres types de MFA, tels que les authentificateurs virtuels et le TOTP matériel

Cas d'utilisation 1 : autoriser uniquement l'enregistrement des appareils certifiés FIPS-140-2 L2

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        }
    ]
}
```

Cas d'utilisation 2 : autoriser l'enregistrement des appareils certifiés FIPS-140-2 L2 et FIDO L1

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
```

```
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2",
                    "iam:FIDO-certification": "L1"
                }
            }
        }
    ]
}
```

Cas d'utilisation 3 : autoriser l'enregistrement des appareils certifiés FIPS-140-2 L2 ou FIPS-140-3 L2

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
```

```
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L2"
                }
            }
        }
    ]
}
```

Cas d'utilisation n°4 : autoriser l'enregistrement des appareils certifiés FIPS-140-2 L2 et prenant en charge d'autres types de MFA, tels que les authentificateurs virtuels et le TOTP matériel

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey": "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey": "Activate",
                    "iam:FIPS-140-2-certification": "L2"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "Null": {
```

```
                    "iam:RegisterSecurityKey": "true"
                }
            }
        }
    ]
}
```

AWS CLI et AWS API

AWS prend en charge l'utilisation de clés d'accès et de clés de sécurité uniquement dans le AWS Management Console. L'utilisation de clés d'accès et de clés de sécurité pour le MFA n'est pas prise en charge dans AWS CLIl'API AWS and, ni pour l'accès aux opérations d'API protégées par le MFA.

Ressources supplémentaires

- Pour plus d'informations sur l'utilisation des clés d'accès et des clés de sécurité dans AWS, consultezActivation d'une clé d'accès ou d'une clé de sécurité (console).
- Pour obtenir de l'aide sur la résolution des problèmes liés aux clés d'accès et aux clés de sécurité AWS, consultezRésolution des problèmes liés aux clés de sécurité FIDO.
- Pour obtenir des informations générales sur le support FIDO2, consultez la rubrique Project FIDO2.

Activation d'un appareil Multi-Factor Authentication (MFA) virtuel (console)

Vous pouvez utiliser un téléphone ou un autre appareil comme appareil d'authentification multifacteur (MFA) virtuel. Pour ce faire, installez une application mobile conforme à RFC 6238, un algorithme TOTP (mot de passe unique basé sur le temps) basé sur des normes. Ces applications génèrent un code d'authentification à six chiffres. Comme ces applications d'authentification multifactorielle (MFA) virtuelle peuvent s'exécuter sur des appareils mobiles non sécurisés, elles peuvent ne pas offrir le même niveau de sécurité que les clés de sécurité FIDO. Nous vous recommandons d'utiliser un dispositif MFA virtuel pendant l'attente de l'approbation d'achat du matériel ou pendant que vous attendez de recevoir votre matériel.

La plupart des applications MFA virtuelles prennent en charge la création de plusieurs appareils virtuels, ce qui vous permet d'utiliser la même application pour plusieurs utilisateurs Comptes AWS . Vous pouvez enregistrer jusqu'à huit appareils MFA de n'importe quelle combinaison des types de MFA actuellement pris en charge auprès de vos Utilisateur racine d'un compte AWS utilisateurs et de ceux d'IAM. Avec plusieurs appareils MFA, vous n'avez besoin que d'un seul appareil MFA pour vous connecter AWS Management Console ou créer une session via cet utilisateur. AWS CLI Nous

vous recommandons d'enregistrer plusieurs appareils MFA. Pour les applications d'authentification, nous recommandons également d'activer la fonctionnalité de sauvegarde ou de synchronisation dans le cloud dans ces applications afin de vous éviter de perdre l'accès à votre compte si vous perdez ou cassez votre appareil avec les applications d'authentification.

Pour obtenir la liste des applications MFA virtuelles que vous pouvez utiliser, consultez Authentification multifactorielle. AWS nécessite une application MFA virtuelle qui génère un code OTP à six chiffres.

Rubriques

- Autorisations nécessaires
- Activation d'un dispositif MFA virtuel pour un utilisateur IAM (Console)
- Remplacer un périphérique MFA virtuel

Autorisations nécessaires

Pour gérer des dispositifs MFA virtuels pour votre utilisateur IAM, vous devez disposer des autorisations de la politique suivante : AWS: permet aux utilisateurs IAM authentifiés MFA de gérer leur propre appareil MFA sur la page des informations d'identification de sécurité.

Activation d'un dispositif MFA virtuel pour un utilisateur IAM (Console)

Vous pouvez utiliser IAM AWS Management Console pour activer et gérer un appareil MFA virtuel pour un utilisateur IAM de votre compte. Vous pouvez attacher des balises à vos ressources IAM, y compris les appareils MFA virtuels, pour les identifier, les organiser et contrôler l'accès. Vous pouvez étiqueter les appareils MFA virtuels uniquement lorsque vous utilisez l'API AWS CLI or AWS . Pour activer et gérer un appareil MFA à l'aide de l' AWS API AWS CLI or, consultez. Activation et gestion des appareils MFA virtuels (AWS CLI ou AWS API) Pour plus d'informations sur le balisage des ressources IAM, consultez Balisage des ressources IAM.

> **Note**
>
> Pour configurer l'authentification MFA, vous devez avoir accès physique au matériel sur lequel le dispositif MFA virtuel de l'utilisateur est hébergé. Par exemple, vous pouvez configurer le MFA pour un utilisateur qui utilisera un dispositif MFA virtuel s'exécutant sur un smartphone. Dans ce cas, vous devez avoir le smartphone à proximité afin de finaliser l'assistant. De ce fait, vous pouvez préférer laisser les utilisateurs configurer et gérer leurs propres dispositifs MFA virtuels. Dans ce cas, vous devez accorder aux utilisateurs

l'autorisation d'exécuter les actions IAM nécessaires. Pour en savoir plus sur la politique IAM qui accorde ces autorisations et pour accéder à un exemple, veuillez consulter la rubrique Didacticiel IAM : permettre aux utilisateurs de gérer leurs informations d'identification et leurs paramètres MFA et la politique d'exemple AWS: permet aux utilisateurs IAM authentifiés MFA de gérer leur propre appareil MFA sur la page des informations d'identification de sécurité.

Pour activer un dispositif MFA virtuel pour un utilisateur IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Dans la liste des Users (Utilisateurs), choisissez le nom d'utilisateur IAM.
4. Choisissez l'onglet Informations d'identification de sécurité. Dans la section Multi-Factor Authentication (MFA) (Authentification multifactorielle (MFA)), sélectionnez Assign MFA device (Attribuer un dispositif MFA).
5. Dans l'assistant, saisissez un nom dans le champ Nom du dispositif, sélectionnez Application Authenticator, puis cliquez sur Suivant.

   IAM génère et affiche les informations de configuration du dispositif MFA virtuel, notamment un graphique de code QR. Le graphique est une représentation de la clé de configuration secrète que l'on peut saisir manuellement sur des périphériques qui ne prennent pas en charge les codes QR.
6. Ouvrez votre application MFA virtuelle. Pour obtenir une liste des applications que vous pouvez utiliser pour héberger des dispositifs MFA virtuels, consultez authentification multifacteur.

   Si l'application MFA virtuelle prend en charge plusieurs comptes ou plusieurs dispositifs MFA virtuels, choisissez l'option permettant de créer un compte ou un dispositif MFA virtuel.
7. Déterminez si l'application MFA prend en charge les codes QR, puis effectuez l'une des actions suivantes :

   - Dans l'assistant, choisissez Show QR code (Afficher le code QR), puis utiliser l'application pour analyser le code QR. Par exemple, vous pouvez choisir l'icône de caméra ou une option similaire à Scan code, puis utiliser la caméra du dispositif pour analyser le code.
   - Dans l'assistant, sélectionnez Show secret key (Afficher la clé secrète), puis saisissez la clé secrète dans votre application MFA.

Une fois que vous avez terminé, le dispositif MFA virtuel commence à générer des mots de passe uniques.

8. Sur la page Configurer l'appareil, accédez à la zone Code MFA 1 et saisissez le mot de passe à usage unique affiché sur le dispositif MFA virtuel. Attendez jusqu'à 30 secondes pour que le dispositif génère un nouveau mot de passe unique. Saisissez ensuite le second mot de passe unique dans la zone MFA Code 2 (Code MFA 2). Choisissez Add MFA (Ajouter un dispositif MFA).

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, le dispositif MFA s'associe avec succès à l'utilisateur mais est désynchronisé. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période. Dans ce cas, vous pouvez resynchroniser le dispositif.

Le périphérique MFA virtuel est désormais prêt à être utilisé avec. AWS Pour plus d'informations sur l'utilisation de la MFA avec le AWS Management Console, consultez. Utilisation de dispositifs MFA avec votre page de connexion IAM

Remplacer un périphérique MFA virtuel

Vous pouvez enregistrer jusqu'à huit appareils MFA de n'importe quelle combinaison des types de MFA actuellement pris en charge auprès de vos Utilisateur racine d'un compte AWS utilisateurs et de ceux d'IAM. Si l'utilisateur perd un périphérique ou a besoin de le remplacer pour une raison quelconque, vous devez désactiver l'ancien périphérique. Vous pouvez alors ajouter le nouveau périphérique pour l'utilisateur.

- Pour désactiver le dispositif actuellement associé à un autre utilisateur IAM, consultez Désactivation des dispositifs MFA.
- Pour ajouter un dispositif MFA virtuel de remplacement pour un autre utilisateur IAM, suivez les étapes de la procédure Activation d'un dispositif MFA virtuel pour un utilisateur IAM (Console) ci-dessus.

- Pour ajouter un périphérique MFA virtuel de remplacement pour le Utilisateur racine d'un compte AWS, suivez les étapes de la procédure. Activation d'un dispositif MFA virtuel pour votre Utilisateur racine d'un compte AWS (console)

Activation d'un jeton TOTP matériel (console)

Un jeton TOTP matériel génère un code numérique à six chiffres basé sur un algorithme TOTP (mot de passe unique à durée limitée). L'utilisateur doit saisir un code valide à partir du périphérique lorsqu'il y est invité lors de la connexion. Chaque dispositif MFA attribué à un utilisateur doit être unique ; un utilisateur ne peut pas saisir un code à partir du périphérique d'un autre utilisateur pour s'authentifier. Les appareils MFA ne peuvent pas être partagés entre comptes ou utilisateurs.

Les jetons TOTP matériels et les clés de sécurité FIDO sont des dispositifs physiques que vous achetez. Les appareils MFA matériels génèrent des codes TOTP pour l'authentification lorsque vous vous connectez à. AWS Ils utilisent des batteries, qui peuvent avoir besoin d'être remplacées et resynchronisées au fil du temps. AWS Les clés de sécurité FIDO, qui utilisent la cryptographie à clé publique, ne nécessitent pas de piles et offrent un processus d'authentification sans faille. Nous vous recommandons d'utiliser les clés de sécurité FIDO pour leur résistance au phishing, ce qui constitue une alternative plus sûre aux appareils TOTP. De plus, les clés de sécurité FIDO peuvent prendre en charge plusieurs utilisateurs IAM ou root sur le même appareil, ce qui améliore leur utilité pour la sécurité des comptes. Pour connaître les spécifications et les informations d'achat de ces deux types d'appareils, consultez authentification multifacteur.

Vous pouvez activer un jeton TOTP matériel pour un utilisateur IAM à partir de la AWS Management Console ligne de commande ou de l'API IAM. Pour activer un dispositif MFA pour votre Utilisateur racine d'un compte AWS, consultez. Activer un jeton TOTP matériel pour Utilisateur racine d'un compte AWS (console)

Vous pouvez enregistrer jusqu'à huit appareils MFA de n'importe quelle combinaison des types de MFA actuellement pris en charge auprès de vos Utilisateur racine d'un compte AWS utilisateurs et de ceux d'IAM. Avec plusieurs appareils MFA, vous n'avez besoin que d'un seul appareil MFA pour vous connecter AWS Management Console ou créer une session via cet utilisateur. AWS CLI

Important

Nous vous recommandons d'activer plusieurs dispositifs MFA pour permettre à vos utilisateurs d'accéder en permanence à votre compte en cas de perte ou d'inaccessibilité d'un dispositif MFA.

Note

Si vous souhaitez activer le dispositif dans la ligne de commande, utilisez `aws iam enable-mfa-device` . Pour activer le dispositif MFA avec l'API IAM, utilisez l'opération `EnableMFADevice`.

Rubriques

- Autorisations nécessaires
- Activation d'un jeton TOTP matériel pour votre propre utilisateur IAM (console)
- Activation d'un jeton TOTP matériel pour un autre utilisateur IAM (console)
- Remplacer un périphérique MFA physique

Autorisations nécessaires

Pour gérer un jeton TOTP matériel pour votre propre utilisateur IAM tout en protégeant les actions sensibles liées à MFA, vous devez disposer des autorisations de la politique suivante :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
```

```
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Activation d'un jeton TOTP matériel pour votre propre utilisateur IAM (console)

Vous pouvez activer votre propre jeton TOTP matériel à partir de la AWS Management Console.

> **Note**
>
> Avant d'activer un jeton TOTP matériel, vous devez y avoir accès physiquement.

Pour activer un jeton TOTP matériel pour votre propre utilisateur IAM (console)

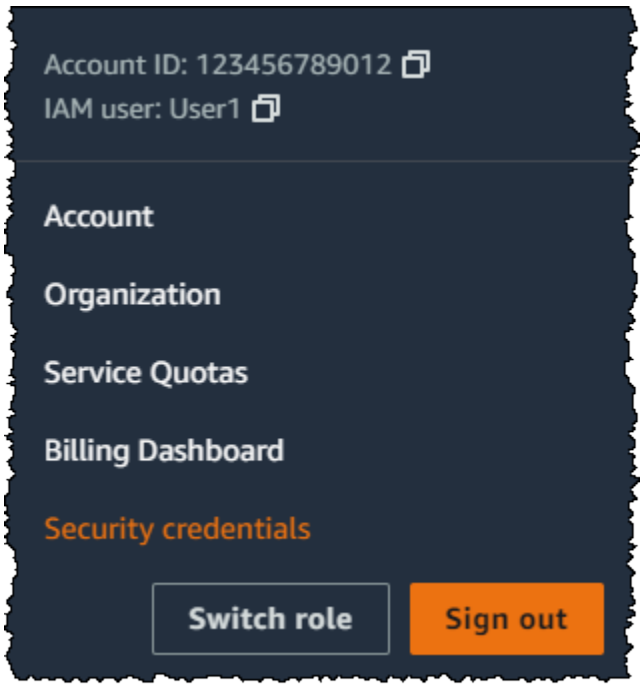
1. Utilisez votre AWS identifiant ou alias de compte, votre nom d'utilisateur IAM et votre mot de passe pour vous connecter à la console IAM.

   > **Note**
   >
   > Pour votre commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser votre nom d'utilisateur IAM et les informations de votre compte. Si vous vous êtes déjà connecté en tant qu'utilisateur différent, sélectionnez Sign in to a different account (Se connecter à un compte différent) en bas de la page pour revenir à la page de connexion principale. À partir de là, vous pouvez saisir votre identifiant de AWS

compte ou votre alias de compte pour être redirigé vers la page de connexion utilisateur IAM de votre compte.

Pour obtenir votre Compte AWS identifiant, contactez votre administrateur.

2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).



3. Dans l'onglet Informations d'identification AWS IAM, sous la section Authentification multifactorielle (MFA), sélectionnez Attribuer le dispositif MFA.
4. Dans l'assistant, tapez le nom du dispositif, choisissez Hardware TOTP token (Jeton TOTP matériel), puis Next (Suivant).
5. Saisissez le numéro de série du périphérique. Le numéro de série se situe généralement l'arrière du périphérique.
6. Dans la zone MFA code 1, saisissez le code à six chiffres qui s'affichent sur le dispositif MFA. Vous devrez peut-être appuyer sur le bouton situé à l'avant du périphérique pour afficher le numéro.

7. Attendez 30 secondes que le périphérique actualise le code, puis saisissez la nouvelle série de six chiffres dans la zone MFA code 2. Vous devrez peut-être appuyer à nouveau sur le bouton situé à l'avant du périphérique pour afficher le second numéro.
8. Choisissez Add MFA (Ajouter un dispositif MFA).

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes d'authentification. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, l'dispositif MFA s'associe avec succès à l'utilisateur mais se désynchronise. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période. Dans ce cas, vous pouvez resynchroniser le dispositif.

L'appareil est prêt à être utilisé avec AWS. Pour plus d'informations sur l'utilisation de l'authentification MFA avec l'interface AWS Management Console, veuillez consulter Utilisation de dispositifs MFA avec votre page de connexion IAM.

Activation d'un jeton TOTP matériel pour un autre utilisateur IAM (console)

Vous pouvez activer un jeton TOTP matériel pour un autre utilisateur IAM à partir de la AWS Management Console.

Pour activer un jeton TOTP matériel pour un autre utilisateur IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Choisissez le nom de l'utilisateur pour lequel vous souhaitez activer l'authentification MFA.
4. Choisissez l'onglet Informations d'identification de sécurité. Dans la section Multi-Factor Authentication (MFA) (Authentification multifactorielle (MFA)), sélectionnez Assign MFA device (Attribuer un dispositif MFA).
5. Dans l'assistant, tapez le nom du dispositif, choisissez Hardware TOTP token (Jeton TOTP matériel), puis Next (Suivant).
6. Saisissez le numéro de série du périphérique. Le numéro de série se situe généralement l'arrière du périphérique.

7. Dans la zone MFA code 1, saisissez le code à six chiffres qui s'affichent sur le dispositif MFA. Vous devrez peut-être appuyer sur le bouton situé à l'avant du périphérique pour afficher le numéro.

8. Attendez 30 secondes que le périphérique actualise le code, puis saisissez la nouvelle série de six chiffres dans la zone MFA code 2. Vous devrez peut-être appuyer à nouveau sur le bouton situé à l'avant du périphérique pour afficher le second numéro.
9. Choisissez Add MFA (Ajouter un dispositif MFA).

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes d'authentification. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, l'dispositif MFA s'associe avec succès à l'utilisateur mais se désynchronise. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période. Dans ce cas, vous pouvez resynchroniser le dispositif.

L'appareil est prêt à être utilisé avec AWS. Pour plus d'informations sur l'utilisation de l'authentification MFA avec l'interface AWS Management Console, veuillez consulter Utilisation de dispositifs MFA avec votre page de connexion IAM.

Remplacer un périphérique MFA physique

Vous pouvez attribuer à un utilisateur jusqu'à huit dispositifs MFA de n'importe quelle combinaison des types MFA actuellement pris en charge, en même temps que vos Utilisateur racine d'un compte AWS utilisateurs et ceux d'IAM. Si l'utilisateur perd un périphérique ou a besoin de le remplacer pour une raison quelconque, vous devez désactiver l'ancien périphérique. Vous pouvez alors ajouter le nouveau périphérique pour l'utilisateur.

- Pour désactiver le périphérique actuellement associé à un utilisateur, consultez la page Désactivation des dispositifs MFA.

- Pour ajouter un jeton TOTP matériel de remplacement pour un utilisateur IAM, suivez les étapes de la procédure Activation d'un jeton TOTP matériel pour un autre utilisateur IAM (console) plus haut dans cette rubrique.
- Pour ajouter un jeton TOTP matériel de remplacement pour le Utilisateur racine d'un compte AWS, suivez les étapes décrites dans la procédure décrite Activer un jeton TOTP matériel pour Utilisateur racine d'un compte AWS (console) plus haut dans cette rubrique.

Activation et gestion des appareils MFA virtuels (AWS CLI ou AWS API)

Vous pouvez utiliser des AWS CLI commandes ou des opérations d' AWS API pour activer un dispositif MFA virtuel pour un utilisateur IAM. Vous ne pouvez pas activer de périphérique MFA Utilisateur racine d'un compte AWS avec l' AWS API AWS CLI, Tools for Windows PowerShell ou tout autre outil de ligne de commande. Toutefois, vous pouvez utiliser le AWS Management Console pour activer un périphérique MFA pour l'utilisateur root.

Lorsque vous activez un dispositif MFA à partir du AWS Management Console, la console exécute plusieurs étapes pour vous. Si vous créez plutôt un appareil virtuel à l' AWS CLI aide des outils pour Windows PowerShell ou de AWS l'API, vous devez effectuer les étapes manuellement et dans le bon ordre. Par exemple, pour créer un dispositif MFA virtuel, vous devez créer l'objet IAM et extraire le code sous forme de chaîne ou de graphique de code QR. Ensuite, vous devez synchroniser le périphérique et l'associer à un utilisateur IAM. Consultez la section Exemples de New-IAMVirtualMFADevice pour plus d'informations. Dans le cas d'un périphérique physique, vous ignorez l'étape de création et passez directement à la synchronisation du périphérique et l'association à un utilisateur.

Vous pouvez attacher des balises à vos ressources IAM, y compris les appareils MFA virtuels, pour les identifier, les organiser et contrôler l'accès. Vous pouvez étiqueter les appareils MFA virtuels uniquement lorsque vous utilisez l'API AWS CLI or AWS .

Un utilisateur IAM utilisant le kit SDK ou l'interface de ligne de commande peut activer un dispositif MFA supplémentaire en appelant `EnableMFADevice` ou désactiver un dispositif MFA existant en appelant `DeactivateMFADevice`. Pour y parvenir, il doit d'abord appeler `GetSessionToken` et soumettre des codes MFA avec un dispositif MFA existant. Cet appel renvoie des informations d'identification de sécurité temporaires qui peuvent ensuite être utilisées pour signer des opérations d'API nécessitant une authentification MFA. Pour un exemple de demande et de réponse, consultez `GetSessionToken` : informations d'identification temporaires pour les utilisateurs qui se trouvent dans des environnements non fiables.

Pour créer l'entité de périphérique virtuel dans IAM pour représenter un dispositif MFA virtuel

Ces commandes fournissent un ARN pour le périphérique qui est utilisé à la place du numéro de série dans un grand nombre des commandes suivantes.

- AWS CLI: `aws iam create-virtual-mfa-device`
- AWS API : `CreateVirtualMFADevice`

Pour activer un périphérique MFA à utiliser avec AWS

Ces commandes synchronisent l'appareil avec un utilisateur AWS et l'associent à celui-ci. S'il s'agit d'un périphérique virtuel, utilisez son ARN en tant que numéro de série.

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes d'authentification. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, l'dispositif MFA s'associe avec succès à l'utilisateur mais se désynchronise. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période. Dans ce cas, vous pouvez resynchroniser l'appareil à l'aide des commandes décrites ci-dessous.

- AWS CLI: `aws iam enable-mfa-device`
- AWS API : `EnableMFADevice`

Pour désactiver un périphérique

Utilisez ces commandes pour dissocier le périphérique de l'utilisateur et le désactiver. S'il s'agit d'un périphérique virtuel, utilisez son ARN en tant que numéro de série. Vous devez également supprimer l'entité de périphérique virtuel séparément.

- AWS CLI: `aws iam deactivate-mfa-device`
- AWS API : `DeactivateMFADevice`

Pour afficher la liste des entités de dispositifs MFA virtuels

Utilisez ces commandes pour afficher la liste des entités de dispositifs MFA virtuels.

- AWS CLI: `aws iam list-virtual-mfa-devices`
- AWS API : `ListVirtualMFADevices`

Pour baliser un appareil MFA virtuel

Utilisez ces commandes pour baliser un appareil MFA virtuel.

- AWS CLI: `aws iam tag-mfa-device`
- AWS API : `TagMFADevice`

Pour répertorier les balises d'un appareil MFA virtuel

Utilisez ces commandes pour répertorier les balises attachées à un appareil MFA virtuel.

- AWS CLI: `aws iam list-mfa-device-tags`
- AWS API : `ListMFADeviceTags`

Pour supprimer la balise d'un appareil MFA virtuel

Utilisez ces commandes pour supprimer les balises attachées à un appareil MFA virtuel.

- AWS CLI: `aws iam untag-mfa-device`
- AWS API : `UntagMFADevice`

Pour resynchroniser un dispositif MFA

Utilisez ces commandes si le périphérique génère des codes qui ne sont pas acceptés par AWS. S'il s'agit d'un périphérique virtuel, utilisez son ARN en tant que numéro de série.

- AWS CLI: `aws iam resync-mfa-device`
- AWS API : `ResyncMFADevice`

Pour supprimer une entité de dispositif MFA virtuel dans IAM

Après avoir dissocié le périphérique de l'utilisateur, vous pouvez supprimer l'entité de périphérique.

- AWS CLI: `aws iam delete-virtual-mfa-device`

- AWS API : `DeleteVirtualMFADevice`

Pour récupérer un dispositif MFA virtuel qui est perdu ou ne fonctionne pas

Parfois, l'appareil d'un utilisateur qui héberge l'application MFA virtuelle est perdu, remplacé ou ne fonctionne pas. Dans ce cas, l'utilisateur ne peut pas le récupérer par lui-même. L'utilisateur doit contacter un administrateur pour désactiver le dispositif. Pour plus d'informations, voir Que faire si un dispositif MFA est perdu ou cesse de fonctionner ?.

Vérification du statut de l'authentification MFA

Utilisez la console IAM pour vérifier si un périphérique MFA valide est activé pour un utilisateur Utilisateur racine d'un compte AWS ou un utilisateur IAM.

Pour vérifier le statut de l'authentification MFA d'un utilisateur racine

1. Connectez-vous à l' AWS Management Console aide de vos informations d'identification d'utilisateur root, puis ouvrez la console IAM à https://console.aws.amazon.com/iam/ l'adresse.
2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).
3. Sous Multi-Factor Authentication (MFA), vérifiez si la MFA est activée ou désactivée. Si la MFA n'a pas été activée, un symbole d'alerte ( ) est affiché.

Si vous souhaitez activer l'authentification MFA pour le compte, consultez l'une des sections suivantes :

- Activation d'un dispositif MFA virtuel pour votre Utilisateur racine d'un compte AWS (console)
- Activer une clé d'accès ou une clé de sécurité pour la Utilisateur racine d'un compte AWS (console)
- Activer un jeton TOTP matériel pour Utilisateur racine d'un compte AWS (console)

Pour vérifier le statut de l'authentification MFA d'utilisateurs IAM

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.

3. Au besoin, ajoutez la colonne MFA à la table des utilisateurs en procédant comme suit :

   a. Au-dessus de la table à l'extrême droite, choisissez l'icône des paramètres ( ).

   b. Dans Manage Columns (Gérer les colonnes), sélectionnez MFA.

   c. (Facultatif) Désactivez la case à cocher des en-têtes de colonnes que vous ne souhaitez pas voir s'afficher dans la table des utilisateurs.

   d. Choisissez Fermer pour revenir à la liste des utilisateurs.

4. La colonne MFA vous fournit des informations sur le dispositif MFA qui est activé. Si aucun dispositif MFA n'est actif pour l'utilisateur, la console affiche None (Aucun). Si l'utilisateur dispose d'un dispositif MFA activé, la colonne MFA affiche le type de dispositif activé avec la valeur de Virtuel, Clé de sécurité, Matériel ou SMS.

   > **Note**
   >
   > AWS fin du support pour l'activation de l'authentification multifactorielle par SMS (MFA). Nous recommandons aux clients dont les utilisateurs IAM utilisent un dispositif MFA basé sur les SMS de passer à l'une des méthodes alternatives suivantes : dispositif MFA virtuel (logiciel), clé de sécurité FIDO ou dispositif MFA matériel. Vous pouvez identifier les utilisateurs dans votre compte avec un dispositif MFA SMS affecté. Pour ce faire, accédez à la console IAM, choisissez Utilisateurs dans le panneau de navigation, puis recherchez les utilisateurs avec SMS mentionné dans la colonne MFA de la table.

5. Pour afficher des informations supplémentaires sur le dispositif MFA pour un utilisateur, choisissez le nom de l'utilisateur dont vous souhaitez vérifier l'état MFA. Choisissez ensuite l'onglet Informations d'identification de sécurité.

6. Si aucun dispositif MFA n'est actif pour l'utilisateur, la console affiche Aucun dispositif MFA. Attribuez un dispositif MFA pour améliorer la sécurité de votre AWS environnement dans la section Authentification à facteurs multiples (MFA). Si les dispositifs MFA de l'utilisateur sont activés, la section Authentification multifactorielle (MFA) affiche des informations détaillées sur les dispositifs :

   - Nom du dispositif
   - Type du dispositif
   - L'identifiant du périphérique, tel que le numéro de série d'un périphérique physique ou l'ARN AWS d'un périphérique virtuel

- Quand le dispositif a été créé

Pour supprimer ou resynchroniser un dispositif, cliquez sur le bouton radio à côté du dispositif, puis choisissez Remove (Supprimer) ou Resync (Resynchroniser).

Pour plus d'informations sur l'activation de l'authentification MFA, consultez la documentation suivante :

- Activation d'un appareil Multi-Factor Authentication (MFA) virtuel (console)
- Activation d'une clé d'accès ou d'une clé de sécurité (console)
- Activation d'un jeton TOTP matériel (console)

Resynchronisation de dispositifs MFA virtuels et matériels

Vous pouvez l'utiliser AWS pour resynchroniser vos appareils d'authentification multifactorielle (MFA) virtuels et matériels. Si votre dispositif n'est pas synchronisé lorsque vous essayez de l'utiliser, la tentative de connexion échoue et IAM vous invite à resynchroniser le dispositif.

> **Note**
>
> Les clés de sécurité FIDO ne se désynchronisent pas. Si une clé de sécurité FIDO est perdue ou endommagée, vous pouvez la désactiver. Pour plus d'informations sur la désactivation de tout type de dispositif MFA, consultez Pour désactiver un dispositif MFA pour un autre utilisateur IAM (console).

En tant qu' AWS administrateur, vous pouvez resynchroniser les appareils MFA virtuels et matériels de vos utilisateurs IAM s'ils ne sont pas synchronisés.

Si votre appareil Utilisateur racine d'un compte AWS MFA ne fonctionne pas, vous pouvez le resynchroniser à l'aide de la console IAM avec ou sans terminer le processus de connexion. Si vous ne parvenez pas à resynchroniser votre appareil, vous devrez peut-être le désassocier et le réassocier. Pour en savoir plus à ce sujet, veuillez consulter les rubriques Désactivation des dispositifs MFA et Activation des appareils MFA pour les utilisateurs de AWS.

Rubriques

- Autorisations nécessaires

- Resynchronisation des dispositifs MFA matériels et virtuels (console IAM)
- Resynchronisation de dispositifs MFA virtuels et matériels (AWS CLI)
- Resynchronisation des périphériques MFA virtuels et matériels (API)AWS

Autorisations nécessaires

Pour resynchroniser des dispositifs MFA virtuels ou matériels pour votre propre utilisateur IAM, vous devez disposer des autorisations de la politique suivante. Cette politique ne vous permet pas de créer ou de désactiver un dispositif.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToViewAndManageTheirOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "BlockAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
```

```
                }
            }
        }
    ]
}
```

Resynchronisation des dispositifs MFA matériels et virtuels (console IAM)

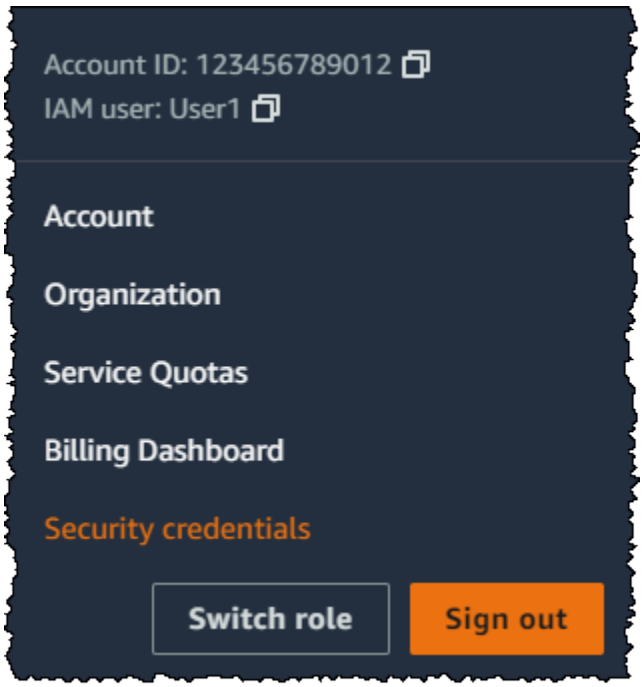
Vous pouvez utiliser la console IAM pour resynchroniser les dispositifs virtuels du matériel MFA.

Pour resynchroniser un dispositif MFA matériel ou virtuel pour votre propre utilisateur IAM (console)

1. Utilisez votre AWS identifiant ou alias de compte, votre nom d'utilisateur IAM et votre mot de passe pour vous connecter à la console IAM.

   > **Note**
   >
   > Pour votre commodité, la page de AWS connexion utilise un cookie de navigateur pour mémoriser votre nom d'utilisateur IAM et les informations de votre compte. Si vous vous êtes déjà connecté en tant qu'utilisateur différent, sélectionnez Sign in to a different account (Se connecter à un compte différent) en bas de la page pour revenir à la page de connexion principale. À partir de là, vous pouvez saisir votre identifiant de AWS compte ou votre alias de compte pour être redirigé vers la page de connexion utilisateur IAM de votre compte.

   Pour obtenir votre Compte AWS identifiant, contactez votre administrateur.
2. Dans la barre de navigation, en haut à droite, choisissez votre nom d'utilisateur, puis Security credentials (Informations d'identification de sécurité).



3. Dans l'onglet Informations d'identification AWS IAM, sous la section Authentification multifactorielle (MFA), cliquez sur le bouton radio situé en regard du dispositif MFA et sélectionnez Resynchroniser.
4. Entrez les deux prochains codes générés séquentiellement à partir du périphérique dans les champs MFA code 1 et MFA code 2. Puis choisissez Resync (Resynchroniser).

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, cette dernière semble fonctionner mais l'appareil reste désynchronisé. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période.

Pour resynchroniser un dispositif MFA matériel ou virtuel pour un autre utilisateur IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Utilisateurs, puis sélectionnez le nom de l'utilisateur dont le dispositif MFA doit être resynchronisé.

3. Choisissez l'onglet Informations d'identification de sécurité. Dans la section Authentification multifactorielle (MFA), cliquez sur le bouton radio situé en regard du dispositif MFA et sélectionnez Resynchroniser.
4. Entrez les deux prochains codes générés séquentiellement à partir du périphérique dans les champs MFA code 1 et MFA code 2. Puis choisissez Resync (Resynchroniser).

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, cette dernière semble fonctionner mais l'appareil reste désynchronisé. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période.

Pour resynchroniser votre dispositif MFA d'utilisateur racine avant la connexion (console)

1. Sur la page Amazon Web Services Sign In With Authentication Device (connexion à Amazon Web Services à l'aide de MFA), choisissez Having problems with your authentication device? (des problèmes avec votre dispositif d'authentification ?) Click here (Cliquez ici).

> **Note**
>
> Il se peut que vous remarquiez des textes différents, tels que se connecter à l'aide de MFA et dépanner votre dispositif d'authentification. Toutefois, les mêmes fonctions sont fournies.

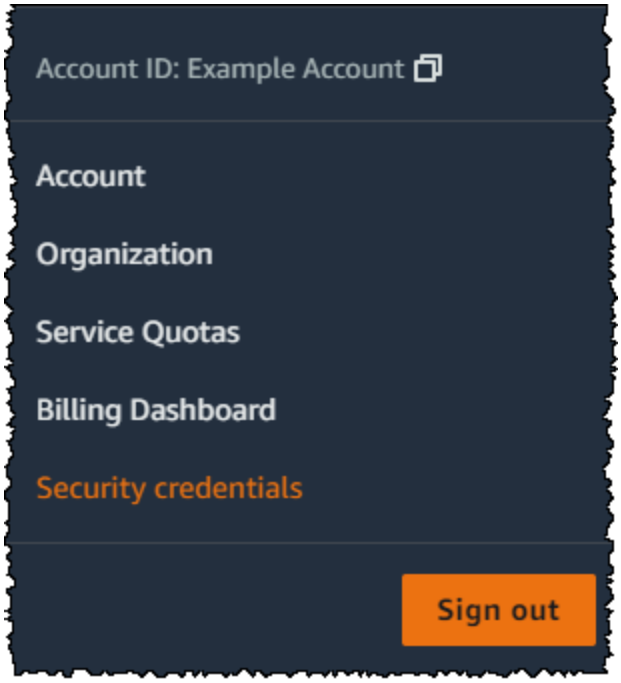
2. Dans la section Re-Sync With Our Servers (Resynchroniser avec nos serveurs), entrez les deux prochains codes générés séquentiellement à partir du périphérique dans les champs MFA code 1 et MFA code 2. Ensuite, choisissez Re-sync authentication device (Resynchroniser l'appareil d'authentification).
3. Si besoin, saisissez à nouveau votre mot de passe et choisissez Sign in (Connexion). Ensuite, procédez à la connexion à l'aide de votre dispositif MFA.

Pour resynchroniser votre dispositif MFA d'utilisateur racine après la connexion (console)

1. Connectez-vous à la console IAM en tant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   > **Note**
   >
   > En tant qu'utilisateur root, vous ne pouvez pas vous connecter à la page Se connecter en tant qu'utilisateur IAM. Si la page Se connecter en tant qu'utilisateur IAM s'affiche, choisissez Se connecter à l'aide de l'adresse e-mail de l'utilisateur root en bas de la page. Pour obtenir de l'aide pour vous connecter en tant qu'utilisateur root, consultez la section Connexion en AWS Management Console tant qu'utilisateur root dans le Guide de Connexion à AWS l'utilisateur.

2. À droite de la barre de navigation, sélectionnez le nom de votre compte, puis Security Credentials (Informations d'identification de sécurité). Au besoin, choisissez Continue to Security credentials (Passer aux informations d'identification de sécurité).



3. Sur la page, développez la section Multi-factor authentication (MFA) (authentification multi-facteur (MFA)).
4. Cliquez sur le bouton radio en regard du dispositif et choisissez Resync (Resynchroniser).

5. Dans la boîte de dialogue Resync MFA device (Resynchroniser le dispositif MFA), entrez les deux prochains codes générés séquentiellement à partir du dispositif dans les champs MFA code 1 et MFA code 2. Puis choisissez Resync (Resynchroniser).

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes. Si vous générez les codes, puis attendez trop longtemps avant d'envoyer la demande, le dispositif MFA s'associe avec succès à l'utilisateur mais est désynchronisé. En effet, les TOTP (Time-based One-Time Passwords ou mots de passe à usage unique à durée limitée) expirent après une courte période.

Resynchronisation de dispositifs MFA virtuels et matériels (AWS CLI)

Vous pouvez resynchroniser les dispositifs MFA virtuels et matériels à partir de l'interface AWS CLI.

Pour resynchroniser un dispositif MFA virtuel ou matériel pour un utilisateur IAM (AWS CLI)

À l'invite de commande, lancez la resync-mfa-device commande aws iam :

- Dispositif MFA virtuel : spécifiez l'Amazon Resource Name (ARN) du périphérique en tant que numéro de série.

```
aws iam resync-mfa-device --user-name Richard --serial-number
 arn:aws:iam::123456789012:mfa/RichardsMFA --authentication-code1 123456 --
authentication-code2 987654
```

- Dispositif MFA matériel : spécifiez le numéro de série du périphérique matériel en tant que numéro de série. Le format est spécifique au fournisseur. Par exemple, vous pouvez acheter un jeton gemalto auprès d'Amazon. Son numéro de série est généralement composé de quatre lettres suivies de quatre chiffres.

```
aws iam resync-mfa-device --user-name Richard --serial-number ABCD12345678 --
authentication-code1 123456 --authentication-code2 987654
```

> **Important**
>
> Envoyez votre demande immédiatement après avoir généré les codes. Si vous générez les codes puis attendez trop longtemps avant d'envoyer la demande, cette dernière échoue car les codes expirent après une courte période.

Resynchronisation des périphériques MFA virtuels et matériels (API)AWS

IAM dispose d'un appel d'API qui effectue la synchronisation. Dans ce cas, nous vous recommandons d'accorder à vos utilisateurs de dispositifs MFA virtuels et matériels l'autorisation d'accès à cet appel d'API. Créez ensuite un outil basé sur cet appel d'API afin que vos utilisateurs puissent resynchroniser leurs périphériques chaque fois que cela est nécessaire.

Pour resynchroniser un périphérique MFA virtuel ou matériel pour un utilisateur IAM (API)AWS

- Envoyez la demande ResyncMFADevice.

Désactivation des dispositifs MFA

Si vous rencontrez des difficultés pour vous connecter avec un dispositif d'authentification multifactorielle (MFA) en tant qu'utilisateur IAM, contactez votre administrateur pour obtenir de l'aide.

En tant qu'administrateur, vous pouvez désactiver le dispositif pour un autre utilisateur IAM. Cela permet à l'utilisateur de se connecter sans utiliser MFA. Vous pouvez faire ceci comme solution provisoire en attendant que le dispositif MFA soit remplacé ou si le périphérique est indisponible temporairement. Par contre, nous vous recommandons d'activer un nouveau périphérique pour l'utilisateur dès que possible. Pour savoir comment activer un nouveau dispositif MFA, consultez the section called “Activation de dispositifs MFA”.

> **Note**
>
> Si vous utilisez l'API ou si vous AWS CLI souhaitez supprimer un utilisateur Compte AWS, vous devez désactiver ou supprimer le dispositif MFA de l'utilisateur. Vous effectuez cette modification dans le cadre du processus de suppression de l'utilisateur. Pour plus d'informations sur la suppression d'utilisateurs, consultez Gestion des utilisateurs IAM.

Rubriques

- Désactivation des dispositifs MFA (console)
- Désactivation des dispositifs MFA (AWS CLI)
- Désactivation des appareils AWS MFA (API)

Désactivation des dispositifs MFA (console)

Pour désactiver un dispositif MFA pour un autre utilisateur IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Pour désactiver le dispositif MFA pour un utilisateur, choisissez le nom de l'utilisateur dont vous souhaitez supprimer le MFA.
4. Choisissez l'onglet Informations d'identification de sécurité.
5. Sous Authentification multifactorielle (MFA), cliquez sur le bouton radio situé en regard du dispositif MFA, puis sélectionnez Supprimer et encore Supprimer.

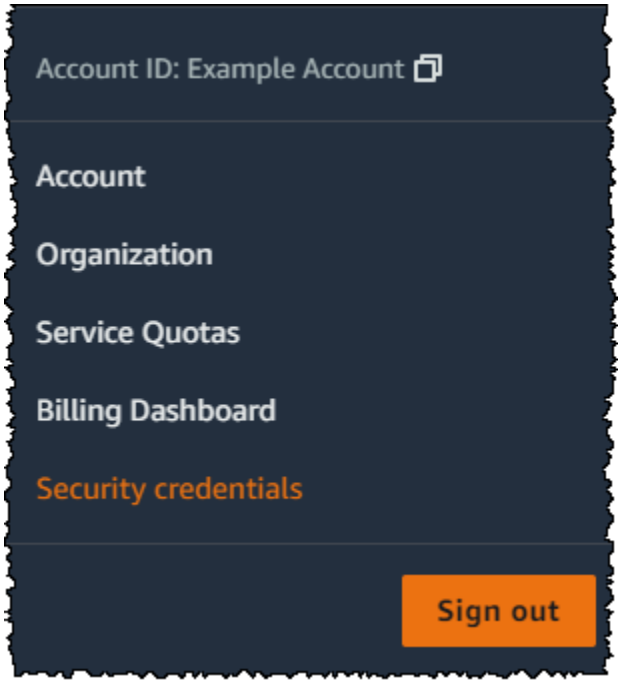
   L'appareil est retiré de AWS. Il ne peut pas être utilisé pour se connecter ou authentifier des demandes tant qu'il n'est pas réactivé et associé à un AWS utilisateur ou. Utilisateur racine d'un compte AWS

Pour désactiver le périphérique MFA pour votre Utilisateur racine d'un compte AWS (console)

1. Connectez-vous à la console IAM en tant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   > **Note**
   >
   > En tant qu'utilisateur root, vous ne pouvez pas vous connecter à la page Se connecter en tant qu'utilisateur IAM. Si la page Se connecter en tant qu'utilisateur IAM s'affiche, choisissez Se connecter à l'aide de l'adresse e-mail de l'utilisateur root en bas de la page. Pour obtenir de l'aide pour vous connecter en tant qu'utilisateur root, consultez la section Connexion en AWS Management Console tant qu'utilisateur root dans le Guide de Connexion à AWS l'utilisateur.

2. À droite de la barre de navigation, sélectionnez le nom de votre compte, puis Security Credentials (Informations d'identification de sécurité). Au besoin, choisissez Continue to Security credentials (Passer aux informations d'identification de sécurité).



3. Dans la section Multi-factor authentication (MFA) (Authentification multifactorielle (MFA)), cliquez sur le bouton radio en regard du dispositif MFA que vous souhaitez désactiver, puis choisissez Remove (Supprimer).
4. Sélectionnez Remove (Supprimer).

   Le dispositif MFA est désactivé pour l' Compte AWS. Vérifiez que l'e-mail qui vous est associé contient Compte AWS un message de confirmation provenant d'Amazon Web Services. L'e-mail vous informe que votre authentification multi-facteur (MFA) Amazon Web Services a été désactivée. Le message viendra de `@amazon.com` ou `@aws.amazon.com`.

Désactivation des dispositifs MFA (AWS CLI)

Pour désactiver un dispositif MFA pour un utilisateur IAM (AWS CLI)

- Exécutez cette commande : `aws iam deactivate-mfa-device`

Désactivation des appareils AWS MFA (API)

Pour désactiver un appareil MFA pour un utilisateur IAM (API)AWS

- Appelez cette opération : `DeactivateMFADevice`

Que faire si un dispositif MFA est perdu ou cesse de fonctionner ?

Si votre périphérique MFA virtuel ou votre jeton TOTP matériel semble fonctionner correctement, mais que vous ne pouvez pas l'utiliser pour accéder à vos AWS ressources, il est possible qu'il ne soit pas synchronisé avec. AWS Pour plus d'informations sur la synchronisation d'un dispositif MFA virtuel ou d'un dispositif MFA matériel, consultez Resynchronisation de dispositifs MFA virtuels et matériels. Les clés de sécurité FIDO ne se désynchronisent pas.

Si votre appareil d' Utilisateur racine d'un compte AWS authentification multifactorielle (MFA) est perdu, endommagé ou ne fonctionne pas, vous pouvez récupérer l'accès à votre compte. Les utilisateurs IAM doivent contacter un administrateur pour désactiver le périphérique.

> **Important**
>
> Nous vous recommandons d'activer plusieurs dispositifs MFA pour permettre à vos utilisateurs IAM d'accéder en permanence à votre compte en cas de perte ou d'inaccessibilité d'un dispositif MFA. Vous pouvez enregistrer jusqu'à huit dispositifs MFA de n'importe quelle combinaison de types MFA actuellement pris en charge avec l'utilisateur root et les utilisateurs IAM de votre Compte AWS .

Récupération d'un dispositif MFA d'utilisateur racine

Si votre appareil d' Utilisateur racine d'un compte AWS authentification multifactorielle (MFA) est perdu, endommagé ou ne fonctionne pas, vous pouvez vous connecter à l'aide d'un autre appareil MFA enregistré sur le même appareil. Utilisateur racine d'un compte AWS Si un seul dispositif MFA est activé pour l'utilisateur root, vous pouvez utiliser d'autres méthodes d'authentification. Cela signifie que si vous ne pouvez pas vous connecter avec votre dispositif MFA, vous pouvez vous connecter en vérifiant votre identité à l'aide de l'e-mail et du numéro de téléphone du contact principal enregistrés avec votre compte.

Avant d'utiliser d'autres facteurs d'authentification pour vous connecter en tant qu'utilisateur root, vous devez pouvoir accéder à l'e-mail et au numéro de téléphone de contact principal qui sont

associés à votre compte. Si vous devez mettre à jour le numéro de téléphone de contact principal, vous pouvez vous connecter en tant qu'utilisateur IAM avec un accès Administrator(Administrateur) au lieu de l'utilisateur root. Pour obtenir des instructions supplémentaires sur la mise à jour des informations de contact du compte, consultez la section Modification de vos informations de contact dans le Guide de l'utilisateur AWS Billing . Si vous n'avez pas accès à une adresse e-mail et à un numéro de téléphone de contact principal, vous devez contacter AWS Support.

> **Important**
>
> Nous vous recommandons de garder à jour l'adresse e-mail et le numéro de téléphone de contact associés à votre utilisateur root pour une restauration réussie du compte. Pour plus d'informations, consultez Mettre à jour le contact principal pour votre Compte AWS dans le guide de référence d'AWS Account Management .

Pour vous connecter en utilisant d'autres facteurs d'authentification en tant que Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.
2. Sur la page Vérification supplémentaire nécessaire, choisissez une méthode MFA pour vous authentifier, puis choisissez Suivant.

   > **Note**
   >
   > Vous pouvez voir un texte de remplacement, tel que Connectez-vous à l'aide de MFA, Dépanner votre dispositif d'authentification, ou Dépanner le MFA, mais les fonctionnalités sont les mêmes. Si vous ne pouvez pas utiliser d'autres facteurs d'authentification pour vérifier l'adresse e-mail et le numéro de téléphone du contact principal de votre compte, contactez AWS Support pour désactiver votre dispositif MFA.

3. Selon le type de MFA que vous utilisez, vous verrez une page différente, mais l'option Dépanner MFA fonctionne de la même manière. Sur la page Vérification supplémentaire nécessaire ou Authentification multifacteur, choisissez Dépanner MFA.
4. Si besoin, saisissez à nouveau votre mot de passe et choisissez Sign in (Connexion).

5. Sur la page Dépanner votre dispositif d'authentification, dans la section Connectez-vous à l'aide d'autres facteurs d'authentification, choisissez Connectez-vous à l'aide d'autres facteurs.
6. Sur la page Connectez-vous à l'aide d'autres facteurs d'authentification, authentifiez votre compte en vérifiant l'adresse e-mail, puis choisissez Envoyer un e-mail de vérification.
7. Vérifiez que l'e-mail associé à votre compte contient un message provenant Compte AWS d'Amazon Web Services (recover-mfa-no-reply@verify .signin.aws). Suivez les instructions de l'e-mail.

   Si vous ne voyez pas d'e-mail dans votre boîte de réception, vérifiez le dossier des courriers indésirables, ou revenez à votre navigateur et choisissez Resend the email (Renvoyer l'e-mail).
8. Une fois que vous avez confirmé votre adresse e-mail, vous pouvez continuer à authentifier votre compte. Pour confirmer votre numéro de téléphone de contact principal, choisissez Call me now (Appelez-moi maintenant).
9. Répondez à l'appel de AWS et, lorsque vous y êtes invité, entrez le numéro à 6 chiffres du AWS site Web sur le clavier de votre téléphone.

   Si vous ne recevez aucun appel de AWS, choisissez Se connecter pour vous reconnecter à la console et recommencer. Ou consultez la rubrique Appareil d'authentification multifactorielle (MFA) perdu ou inutilisable pour obtenir l'aide du support.
10. Une fois que vous avez confirmé votre numéro de téléphone, vous pouvez vous connecter à votre compte en choisissant Sign in to the console (Se connecter à la console).
11. L'étape suivante varie selon le type de MFA que vous utilisez :

    - Si vous utilisez un dispositif MFA virtuel, supprimez le compte de votre périphérique. Ensuite, accédez à la page Informations d'identification de sécuritéAWS et supprimez l'ancienne entité de dispositif virtuel MFA avant d'en créer une nouvelle.
    - Pour une clé de sécurité FIDO, accédez à la page Informations d'identification de sécuritéAWS et désactivez l'ancienne clé FIDO avant d'en activer une nouvelle.
    - Si vous utilisez un jeton TOTP matériel, contactez le fournisseur tiers pour qu'il dépanne ou remplace le dispositif. Vous pouvez continuer à vous connecter à l'aide d'autres facteurs d'authentification jusqu'à ce que vous receviez votre nouveau périphérique. Une fois que vous avez le nouveau dispositif MFA matériel, rendez-vous sur la page Informations d'identification de sécuritéAWS et supprimez l'ancienne entité de dispositif matériel MFA avant d'en créer une nouvelle.

> **Note**
>
> Vous n'êtes pas obligé de remplacer un dispositif MFA perdu ou volé par le même type de périphérique. Par exemple, si vous cassez votre clé de sécurité FIDO et en commandez une nouvelle, vous pouvez utiliser un dispositif MFA virtuel ou un jeton TOTP matériel jusqu'à ce que vous receviez la nouvelle clé de sécurité FIDO.

> **Important**
>
> Si votre dispositif MFA est manquant ou volé, après vous être connecté à l'aide d'autres modes d'authentification et avoir mis en place votre dispositif MFA de remplacement, modifiez votre mot de passe d'utilisateur root au cas où un pirate aurait volé le dispositif d'authentification et pourrait également détenir votre mot de passe actuel. Pour plus d'informations, consultez Modifier le mot de passe Utilisateur racine d'un compte AWSdans le AWS Account Management Guide de référence.

Récupération d'un dispositif MFA d'utilisateur IAM

Si vous êtes un utilisateur IAM et que votre périphérique est perdu ou a cessé de fonctionner, vous ne pouvez pas le récupérer vous-même. Vous devez contacter un administrateur pour désactiver le périphérique. Ensuite, vous pouvez activer un nouveau périphérique.

Pour obtenir de l'aide pour un dispositif MFA en tant qu'utilisateur IAM

1. Contactez l' AWS administrateur ou toute autre personne qui vous a fourni le nom d'utilisateur et le mot de passe de l'utilisateur IAM. L'administrateur doit désactiver le dispositif MFA, comme décrit dans Désactivation des dispositifs MFA, afin que vous puissiez vous connecter.
2. L'étape suivante varie selon le type de MFA que vous utilisez :

   - Si vous utilisez un dispositif MFA virtuel, supprimez le compte de votre périphérique. Activez ensuite le périphérique virtuel comme décrit dans Activation d'un appareil Multi-Factor Authentication (MFA) virtuel (console).

- Si vous utilisez une clé de sécurité FIDO, contactez le fournisseur tiers pour qu'il remplace le périphérique. Lorsque vous recevez la nouvelle clé de sécurité FIDO, activez-la comme décrit dans Activation d'une clé d'accès ou d'une clé de sécurité (console).
- Si vous utilisez un jeton TOTP matériel, contactez le fournisseur tiers pour qu'il dépanne ou remplace le dispositif. Une fois que vous avez le nouveau dispositif MFA, activez-le comme décrit dans Activation d'un jeton TOTP matériel (console).

> **Note**
>
> Vous n'êtes pas obligé de remplacer un dispositif MFA perdu ou volé par le même type de périphérique. Vous pouvez avoir jusqu'à huit dispositifs MFA, quelle que soit leur combinaison. Par exemple, si vous cassez votre clé de sécurité FIDO et en commandez une nouvelle, vous pouvez utiliser un dispositif MFA virtuel ou un jeton TOTP matériel jusqu'à ce que vous receviez la nouvelle clé de sécurité FIDO.

3. Si votre dispositif MFA est perdu ou volé, modifiez votre mot de passe , au cas où un pirate informatique aurait volé le dispositif d'authentification et détiendrait également votre mot de passe actuel. Pour de plus amples informations, veuillez consulter Gestion des mots de passe des utilisateurs IAM.

Configuration de l'accès aux API protégé par MFA

Avec les politiques IAM, vous pouvez spécifier quelles opérations d'API un utilisateur est autorisé à appeler. Dans certains cas, vous souhaiterez peut-être renforcer la sécurité en imposant aux utilisateurs de s'authentifier à l'aide de l'authentification multi-facteur (MFA) AWS avant qu'ils soient autorisés à exécuter des actions particulièrement sensibles.

Par exemple, vous disposez peut-être d'une politique qui autorise l'utilisateur à exécuter les actions Amazon EC2 `RunInstances`, `DescribeInstances` et `StopInstances`. Mais vous souhaiterez peut-être restreindre une action destructrice telle que `TerminateInstances` et vous assurer que les utilisateurs ne peuvent effectuer cette action que s'ils s'authentifient auprès d'un périphérique AWS MFA.

Rubriques

- Présentation
- Scénario : protection MFA pour la délégation entre comptes

- Scénario : protection MFA pour l'accès aux opérations d'API du compte actuel
- Scénario : protection MFA des ressources disposant de politiques basées sur les ressources

Présentation

L'ajout d'une protection MFA aux opérations d'API nécessite les tâches suivantes :

1. L'administrateur configure un dispositif AWS MFA pour chaque utilisateur qui doit effectuer des demandes d'API nécessitant une authentification MFA. Ce processus est décrit dans la section Activation des appareils MFA pour les utilisateurs de AWS.
2. L'administrateur crée des politiques pour les utilisateurs qui incluent un `Condition` élément qui vérifie si l'utilisateur s'est authentifié avec un dispositif AWS MFA.
3. L'utilisateur appelle l'une des opérations d' AWS STS API prenant en charge les paramètres MFA AssumeRoleou GetSessionToken, selon le scénario de protection MFA, comme expliqué plus loin. Dans le cadre de l'appel, l'utilisateur inclut l'identifiant du périphérique associé à l'utilisateur. L'utilisateur inclut également le mot de passe TOTP (Time-based One-Time Password) que le périphérique génère. Dans tous les cas, l'utilisateur récupère les informations d'identification de sécurité temporaires qu'il peut ensuite utiliser pour effectuer des demandes supplémentaires à AWS.

> **Note**
>
> La protection MFA des opérations d'API d'un service est uniquement disponible si le service prend en charge les informations d'identification de sécurité temporaires. Pour connaître la liste de ces services, veuillez consulter Utilisation d'informations d'identification de sécurité temporaires pour accéder à AWS.

Si l'autorisation échoue, AWS renvoie un message d'erreur de refus d'accès (comme c'est le cas pour tout accès non autorisé). Lorsque des politiques d'API protégées par MFA sont en place, AWS refuse l'accès aux opérations d'API spécifiées dans les politiques si l'utilisateur tente d'appeler une opération d'API sans authentification MFA valide. L'opération est également refusée si l'horodatage de la demande pour l'opération d'API se situe en dehors de la plage autorisée spécifiée dans la politique. L'utilisateur doit être à nouveau authentifié avec l'authentification MFA en demandant de nouvelles informations d'identification de sécurité temporaires avec un code MFA et un numéro de série de périphérique.

Politiques IAM avec des conditions MFA

Les politiques avec des conditions MFA peuvent être attachées à ce qui suit :

- Un utilisateur ou un groupe IAM
- Une ressource telle qu'un compartiment Amazon S3, une file d'attente Amazon SQS ou une rubrique Amazon SNS
- La politique d'approbation d'un rôle IAM qui peut être endossée par un utilisateur

Vous pouvez utiliser une condition MFA d'une politique pour vérifier les propriétés suivantes :

- Existence : pour vérifier simplement que l'utilisateur s'est authentifié par MFA, vérifiez que la clé `aws:MultiFactorAuthPresent` est `True` dans une condition `Bool`. La clé est uniquement présente lorsque l'utilisateur s'authentifie avec des informations d'identification à court terme. Les informations d'identification à long terme, telles que les clés d'accès, n'incluent pas cette clé.
- Durée : si vous voulez octroyer l'accès uniquement dans un délai spécifié après l'authentification MFA, utilisez un type de condition numérique pour comparer l'âge de la clé `aws:MultiFactorAuthAge` à une valeur (par exemple 3 600 secondes). Notez que la clé `aws:MultiFactorAuthAge` n'est pas présente si l'authentification MFA n'a pas été utilisée.

L'exemple suivant présente la politique d'approbation d'un rôle IAM incluant une condition MFA pour tester l'existence de l'authentification MFA. Avec cette politique, les utilisateurs Compte AWS spécifiés dans l'`Principal`élément (à `ACCOUNT-B-ID` remplacer par un Compte AWS identifiant valide) peuvent assumer le rôle auquel cette politique est attachée. Toutefois, ces utilisateurs peuvent endosser le rôle uniquement s'ils se sont authentifiés à l'aide de l'authentification MFA.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "ACCOUNT-B-ID"},
    "Action": "sts:AssumeRole",
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }
}
```

Pour plus d'informations sur les types de conditions pour l'authentification MFA, consultez AWS clés contextuelles de condition globale, Opérateurs de condition numériques et Opérateur de condition pour vérifier l'existence de clés de condition

Choisir entre GetSessionToken et AssumeRole

AWS STS fournit deux opérations d'API qui permettent aux utilisateurs de transmettre des informations MFA : `GetSessionToken` et. `AssumeRole` L'opération d'API que l'utilisateur appelle pour obtenir des informations d'identification de sécurité temporaires dépend du scénario applicable parmi les suivants :

Utilisez **`GetSessionToken`** pour les scénarios suivants :

- Appelez les opérations d'API qui accèdent aux ressources de la même manière Compte AWS que l'utilisateur IAM qui fait la demande. Notez que les informations d'identification temporaires issues d'une `GetSessionToken` demande ne peuvent accéder aux opérations IAM et AWS STS API que si vous incluez des informations MFA dans la demande d'informations d'identification. Du fait que les informations d'identification temporaires renvoyées par `GetSessionToken` incluent des informations d'authentification MFA, vous pouvez vérifier l'authentification MFA dans les opérations d'API individuelles effectuées par les informations d'identification.
- Accédez aux ressources protégées par des politiques basées sur les ressources qui incluent une condition MFA.

Le but principal de l'opération `GetSessionToken` est d'authentifier l'utilisateur à l'aide de l'authentification MFA. Vous ne pouvez pas utiliser de stratégies pour contrôler les opérations d’authentification.

Utilisez **`AssumeRole`** pour les scénarios suivants :

- Appeler des opérations d'API qui accèdent à des ressources dans le même Compte AWS ou dans un compte différent. Les appels d'API peuvent inclure n'importe quel IAM ou AWS STS API. Notez que pour protéger l'accès vous devez appliquer l'authentification MFA au moment où l'utilisateur endosse le rôle. Les informations d'identification temporaires renvoyées par `AssumeRole` n'incluent pas d'informations d'authentification MFA dans ce contexte ; vous ne pouvez donc pas vérifier les opérations d'API individuelles pour l'authentification MFA. C'est pourquoi vous devez utiliser `GetSessionToken` pour restreindre l'accès aux ressources protégées par des politiques basées sur les ressources.

Vous trouverez des détails sur l'implémentation de ces scénarios ultérieurement dans ce document.

Points importants sur l'accès aux API protégé par MFA

Il est important de comprendre les aspects suivants relatifs à la protection MFA pour les opérations d'API :

- La protection MFA est uniquement disponible avec les informations d'identification de sécurité temporaires que vous devez obtenir avec `AssumeRole` ou `GetSessionToken`.
- Vous ne pouvez pas utiliser l'accès à l'API protégé par MFA avec des informations d'identification. Utilisateur racine d'un compte AWS
- Vous ne pouvez pas utiliser l'accès aux API protégé par MFA avec les clés de sécurité U2F.
- Les utilisateurs fédérés ne peuvent pas se voir attribuer de périphérique MFA à utiliser AWS avec les services. Ils ne peuvent donc pas AWS accéder aux ressources contrôlées par le MFA. (Voir le point suivant.)
- Les autres opérations d' AWS STS API qui renvoient des informations d'identification temporaires ne prennent pas en charge le MFA. Pour `AssumeRoleWithWebIdentity` et`AssumeRoleWithSAML`, l'utilisateur est authentifié par un fournisseur externe et AWS ne peut pas déterminer si ce fournisseur a besoin du MFA. Pour `GetFederationToken`, l'authentification MFA n'est pas nécessairement associée à un utilisateur spécifique.
- De même, les informations d'identification à long terme (clés d'accès utilisateur IAM et clés d'accès de l’utilisateur racine) ne peuvent pas être utilisées avec l'accès aux API protégé par MFA, car elles n'expirent pas.
- `AssumeRole` et `GetSessionToken` peuvent également être appelées sans informations MFA. Dans ce cas, le principal récupère les informations d'identification de sécurité temporaires, mais les informations de session de ces informations d'identification temporaires n'indiquent pas que l'utilisateur s'est authentifié avec l'authentification MFA.
- Pour établir la protection MFA pour les opérations d'API, ajoutez des conditions d'authentification MFA aux politiques. Une politique doit inclure la clé de condition `aws:MultiFactorAuthPresent` pour imposer l'utilisation de l'authentification MFA. Pour la délégation entre comptes, la politique d'approbation du rôle doit inclure la clé de condition.
- Lorsque vous autorisez une autre personne Compte AWS à accéder aux ressources de votre compte, la sécurité de vos ressources dépend de la configuration du compte approuvé (l'autre compte, pas le vôtre). Cela est vrai même lorsque vous imposez une authentification multi-facteur. Toutes les identités du compte approuvé ayant l'autorisation de créer des dispositifs MFA virtuels peut créer une demande de MFA pour satisfaire cette partie de la politique d'approbation de votre

rôle. Avant d'autoriser les membres d'un autre compte à accéder à vos AWS ressources qui nécessitent une authentification à plusieurs facteurs, vous devez vous assurer que le propriétaire du compte de confiance suit les meilleures pratiques en matière de sécurité. Par exemple, le compte approuvé doit restreindre l'accès aux opérations d'API sensibles, telles que les opérations d'API de gestion de dispositif MFA, à des identités approuvées spécifiques.

- Si une politique inclut une condition MFA, une demande est refusée si les utilisateurs n'ont pas été authentifiés par MFA ou s'ils fournissent un identifiant de dispositif MFA ou un TOTP non valide.

Scénario : protection MFA pour la délégation entre comptes

Dans ce scénario, vous souhaitez déléguer l'accès aux utilisateurs IAM d'un autre compte, mais uniquement si les utilisateurs sont authentifiés à l'aide d'un appareil MFA AWS . (Pour plus d'informations sur la délégation entre comptes, consultez Termes et concepts relatifs aux rôles.

Imaginons que vous disposiez d'un compte A (le compte d'approbation qui est titulaire de la ressource auxquelles les utilisateurs ont accès), avec l'utilisateur IAM Anaya qui dispose de l'autorisation administrateur. Elle souhaite accorder l'accès à l'utilisateur Richard au compte B (le compte approuvé), mais veut s'assurer que Richard s'est authentifié avec MFA avant d'endosser le rôle.

1. Dans le compte de confiance A, Anaya crée un rôle IAM nommé `CrossAccountRole` et définit le principal de la politique de confiance du rôle sur l'ID du compte B. La politique de confiance autorise l'action. AWS STS `AssumeRole` Anaya ajoute également une condition MFA à la politique d'approbation, comme dans l'exemple suivant.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "ACCOUNT-B-ID"},
    "Action": "sts:AssumeRole",
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }
}
```

2. Anaya ajoute une politique d'autorisations au rôle qui spécifie ce que le rôle est autorisé à faire. La politique d'autorisations d'un rôle avec protection MFA est identique à toutes les politiques d'autorisation de rôle. L'exemple suivant montre la politique qu'Anaya ajoute au rôle : elle permet à un utilisateur endossant le rôle d'exécuter toutes les actions Amazon DynamoDB sur la table

`Books` dans le compte A. Cette politique permet également l'action `dynamodb:ListTables`, qui est nécessaire pour effectuer des actions dans la console.

> **Note**
>
> La politique d'autorisations n'inclut pas de condition MFA. Il est important de comprendre que l'authentification MFA sert uniquement à déterminer si un utilisateur peut endosser le rôle. Une fois que l'utilisateur endosse le rôle, aucune autre vérification MFA n'est effectuée.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TableActions",
            "Effect": "Allow",
            "Action": "dynamodb:*",
            "Resource": "arn:aws:dynamodb:*:ACCOUNT-A-ID:table/Books"
        },
        {
            "Sid": "ListTables",
            "Effect": "Allow",
            "Action": "dynamodb:ListTables",
            "Resource": "*"
        }
    ]
}
```

3. Dans le compte sécurisé B, l'administrateur s'assure que l'utilisateur IAM Richard est configuré avec un appareil AWS MFA et qu'il connaît l'identifiant de l'appareil. L'ID du périphérique est le numéro de série s'il s'agit d'un dispositif MFA matériel, ou l'ARN du périphérique s'il s'agit d'un dispositif MFA virtuel.
4. Dans le compte B, l'administrateur attache la politique suivante à l'utilisateur Richard (ou à un groupe dont il est membre) qui l'autorise à appeler l'action `AssumeRole`. La ressource est définie sur l'ARN du rôle qu'Anaya a créé à l'étape 1. Notez que cette politique n'inclut aucune condition MFA.

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [{
    "Effect": "Allow",
    "Action": ["sts:AssumeRole"],
    "Resource": ["arn:aws:iam::ACCOUNT-A-ID:role/CrossAccountRole"]
  }]
}
```

5. Dans le compte B, Richard (ou une application qu'il exécute) appelle `AssumeRole`. L'appel d'API inclut l'ARN du rôle à endosser (`arn:aws:iam::ACCOUNT-A-ID:role/CrossAccountRole`), l'ID du dispositif MFA et le TOTP actuel que Richard obtient de ce périphérique.

   Lorsque Richard appelle`AssumeRole`, il AWS détermine s'il possède des informations d'identification valides, y compris l'exigence du MFA. Le cas échéant, Richard réussit à endosser le rôle et peut exécuter n'importe quelle action DynamoDB sur la table nommée `Books` dans le compte A tout en utilisant les informations d'identification temporaires du rôle.

   Pour obtenir un exemple d'un programme qui appelle `AssumeRole`, consultez Appels AssumeRole avec authentification MFA.

Scénario : protection MFA pour l'accès aux opérations d'API du compte actuel

Dans ce scénario, vous devez vous assurer qu'un utilisateur Compte AWS peut accéder aux opérations d'API sensibles uniquement lorsqu'il est authentifié à l'aide d'un dispositif AWS MFA.

Imaginons que vous ayez un compte A contenant un groupe de développeurs ayant besoin d'utiliser des instances EC2. Les développeurs standard peuvent utiliser les instances, mais ils n'ont pas l'autorisation d'utiliser les actions `ec2:StopInstances` ou `ec2:TerminateInstances`. Vous souhaitez limiter ces actions « destructives » réalisées avec des actions à quelques utilisateurs approuvés uniquement, vous ajoutez donc la protection MFA à la politique qui autorise ces actions Amazon EC2 sensibles.

Dans ce scénario, l'un des utilisateurs approuvé s'appelle Sofía. L'utilisatrice Anaya est administratrice du compte A.

1. Anaya s'assure que Sofía est configurée avec un appareil AWS MFA et que Sofía connaît l'identifiant de l'appareil. L'ID du périphérique est le numéro de série s'il s'agit d'un dispositif MFA matériel, ou l'ARN du périphérique s'il s'agit d'un dispositif MFA virtuel.
2. Anaya crée un groupe appelé `EC2-Admins` et ajoute l'utilisatrice Sofía au groupe.

3. Anaya attache la politique suivante au groupe `EC2-Admins`. Cette politique autorise également les utilisateurs à appeler les actions `StopInstances` et `TerminateInstances` Amazon EC2 uniquement si l'utilisateur s'est authentifié à l'aide de l'authentification MFA.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "ec2:StopInstances",
      "ec2:TerminateInstances"
    ],
    "Resource": ["*"],
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

4.

> **Note**
>
> Pour que cette politique prenne effet, les utilisateurs doivent d'abord se déconnecter et se connecter à nouveau.

   Si l'utilisatrice Sofía a besoin d'arrêter ou de résilier une instance Amazon EC2, elle (ou une application qu'elle exécute) appelle `GetSessionToken`. Cette opération d'API transmet l'ID du dispositif MFA et le TOTP actuel que Sofía obtient de son périphérique.

5. L'utilisatrice Sofía (ou une application qu'elle utilise) utilise les informations d'identification temporaires fournies par `GetSessionToken` pour appeler l'action `StopInstances` ou `TerminateInstances` Amazon EC2.

   Pour obtenir un exemple d'un programme qui appelle `GetSessionToken`, consultez Appels GetSessionToken avec authentification MFA ci-après dans ce document.

Scénario : protection MFA des ressources disposant de politiques basées sur les ressources

Dans ce scénario, vous êtes le propriétaire d'un compartiment S3, d'une file d'attente SQS ou d'une rubrique SNS. Vous devez vous assurer que tous les Compte AWS utilisateurs accédant à la ressource sont authentifiés par un dispositif MFA AWS .

Ce scénario illustre un processus permettant de fournir une protection MFA entre comptes sans nécessiter que les utilisateurs endossent d'abord un rôle. Dans ce cas, l'utilisateur peut accéder à la ressource s'il répond à trois conditions : l'utilisateur doit être authentifié par l'authentification MFA, être en mesure d'obtenir des informations d'identification temporaires de `GetSessionToken` et disposer d'un compte approuvé par la politique de la ressource.

Imaginons que vous vous trouvez dans le compte A et que vous créez un compartiment S3. Vous souhaitez accorder l'accès à ce compartiment à des utilisateurs appartenant à plusieurs entités différentes Comptes AWS, mais uniquement s'ils sont authentifiés à l'aide de la MFA.

Dans ce scénario, l'utilisatrice Anaya est une administratrice du compte A. L'utilisateur Nikhil est un utilisateur IAM du compte C.

1. Dans le compte A, Anaya crée un compartiment appelé `Account-A-bucket`.
2. Anaya ajoute la politique de compartiment au compartiment. La politique autorise n'importe quel utilisateur du compte A, du compte B ou du compte C à exécuter les actions `PutObject` et `DeleteObject` Amazon S3 dans le compartiment. La politique inclut une condition MFA.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"AWS": [
      "ACCOUNT-A-ID",
      "ACCOUNT-B-ID",
      "ACCOUNT-C-ID"
    ]},
    "Action": [
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": ["arn:aws:s3:::ACCOUNT-A-BUCKET-NAME/*"],
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

> **Note**
>
> Amazon S3 fournit une fonction de suppression MFA (MFA Delete) pour l'accès au compte racine (uniquement). Vous pouvez activer la fonction de suppression MFA d'Amazon S3

lorsque vous définissez l'état de la gestion des versions du compartiment. La fonction de suppression MFA d'Amazon S3 ne peut pas être appliquée à un utilisateur IAM et elle est gérée indépendamment de l'accès à l'API protégé par MFA. Un utilisateur IAM disposant des autorisations nécessaires pour supprimer un compartiment ne peut pas le faire si la fonction de suppression MFA Amazon S3 est activée. Pour plus d'informations sur la fonction Supprimer MFA d'Amazon S3, consultez la section Supprimer MFA.

3. Dans le compte C, un administrateur vérifie que l'utilisateur Nikhil est configuré avec un dispositif MFA AWS et qu'il connaît l'ID du dispositif. L'ID du périphérique est le numéro de série s'il s'agit d'un dispositif MFA matériel, ou l'ARN du périphérique s'il s'agit d'un dispositif MFA virtuel.
4. Dans le compte C, Nikhil (ou une application qu'il exécute) appelle `GetSessionToken`. L'appel inclut l'ID ou l'ARN du dispositif MFA et le TOTP actuel que Nikhil obtient de ce périphérique.
5. Nikhil (ou une application qu'il utilise) utilise les informations d'identification temporaires renvoyées par `GetSessionToken` pour appeler l'action `PutObject` Amazon S3 afin de télécharger un fichier dans `Account-A-bucket`.

   Pour obtenir un exemple d'un programme qui appelle `GetSessionToken`, consultez Appels GetSessionToken avec authentification MFA ci-après dans ce document.

> **Note**
>
> Les informations d'identification temporaires renvoyées par `AssumeRole` ne fonctionnent pas dans ce cas. Bien que l'utilisateur puisse fournir les informations MFA lui permettant d'endosser un rôle, les informations d'identification temporaires renvoyées par `AssumeRole` n'incluent par les informations MFA. Ces informations sont requises pour satisfaire la condition MFA de la politique.

Exemple de code : demande d'informations d'identification avec l'authentification multifacteur

Les exemples suivants montrent comment appeler les opérations `GetSessionToken` et `AssumeRole` et transmettre les paramètres d'authentification MFA. Aucune autorisation n'est requise pour appeler `GetSessionToken`, toutefois, vous devez disposer d'une politique vous permettant d'appeler `AssumeRole`. Les informations d'identification renvoyées sont ensuite utilisées afin de répertorier tous les compartiments S3 dans le compte.

Appels GetSessionToken avec authentification MFA

L'exemple suivant explique comment appeler `GetSessionToken` et transmettre les informations d'authentification MFA. Les informations d'identification de sécurité temporaires renvoyées par l'opération `GetSessionToken` sont ensuite utilisées pour répertorier tous les compartiments S3 dans le compte.

La politique associée à l'utilisateur exécutant ce code (ou à un groupe dont fait partie l'utilisateur) fournit les autorisations pour les informations d'identification temporaires renvoyées. Dans cet exemple de code, la politique doit autoriser l'utilisateur à demander l'opération `ListBuckets` Amazon S3.

Les exemples de code suivants montrent comment utiliser`GetSessionToken`.

CLI

AWS CLI

Pour obtenir un ensemble d’informations d’identification à court terme d’une identité IAM

La commande `get-session-token` suivante extrait un ensemble d’informations d’identification à court terme pour l’identité IAM qui effectue l’appel. Les informations d’identification obtenues peuvent être utilisées pour les requêtes où l’authentification multifactorielle (MFA) est requise par la politique. Les informations d’identification expirent 15 minutes après leur création.

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

Sortie :

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
```

```
    }
}
```

Pour plus d'informations, consultez Demande d'informations d'identification temporaires de sécurité dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetSessionTokenà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : renvoie une **`Amazon.RuntimeAWSCredentials`** instance contenant des informations d'identification temporaires valides pendant une période définie. Les informations d'identification utilisées pour demander des informations d'identification temporaires sont déduites des paramètres par défaut actuels du shell. Pour spécifier d'autres informations d'identification, utilisez les SecretKey paramètres - ProfileName ou - AccessKey /-.

```
Get-STSSessionToken
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemple 2 : renvoie une **`Amazon.RuntimeAWSCredentials`** instance contenant des informations d'identification temporaires valides pendant une heure. Les informations d'identification utilisées pour effectuer la demande sont obtenues à partir du profil spécifié.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
```

```
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemple 3 : renvoie une **`Amazon.RuntimeAWSCredentials`** instance contenant des informations d'identification temporaires valides pendant une heure en utilisant le numéro d'identification de l'appareil MFA associé au compte dont les informations d'identification sont spécifiées dans le profil « myprofilename » et la valeur fournie par l'appareil.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- Pour plus de détails sur l'API, reportez-vous GetSessionTokenà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Obtenez un jeton de session en transmettant un jeton MFA et utilisez-le pour répertorier les compartiments Amazon S3 pour le compte.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
```

```
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Pour plus de détails sur l'API, consultez GetSessionTokenle AWS manuel de référence de l'API SDK for Python (Boto3).

Appels AssumeRole avec authentification MFA

Les exemples suivants, expliquent comment appeler `AssumeRole` et transmettre les informations d'authentification MFA. Les informations d'identification de sécurité temporaires renvoyées par `AssumeRole` sont ensuite utilisées pour répertorier tous les compartiments Amazon S3 du compte.

Pour plus d'informations sur ce scénario, consultez Scénario : protection MFA pour la délégation entre comptes.

Les exemples de code suivants montrent comment utiliser`AssumeRole`.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
```

```
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);

            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
```

```
    }
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK for .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
#############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function sts_assume_role
#
```

```
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
################################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
        ech o"Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
```

```
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
                             Aws::Auth::AWSCredentials &credentials,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;
```

```
    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour endosser un rôle

La commande `assume-role` suivante extrait un ensemble d'informations d'identification à court terme pour le rôle IAM `s3-access-example`.

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
```

```
    --role-session-name s3-access-example
```

Sortie :

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

La sortie de la commande contient une clé d’accès, une clé secrète et un jeton de session que vous pouvez utiliser pour vous authentifier auprès d’ AWS.

Pour l'utilisation de la AWS CLI, vous pouvez configurer un profil nommé associé à un rôle. Lorsque vous utilisez le profil, la AWS CLI appelle assume-role et gère les informations d'identification pour vous. Pour plus d'informations, consultez la section Utiliser un rôle IAM dans la AWS CLI dans le Guide de l'utilisateur de la AWS CLI.

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version": "2012-10-17",
 * "Statement": [
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code
 * example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
```

```
 * Role" in the AWS Directory Service guide.
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, rn:aws:iam::000008047983:role/s3role).\s
                    roleSessionName - An identifier for the assumed role session
 (for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
 String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
```

```
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
 DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
 exTime);

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez le client.

```
import { STSClient } from "@aws-sdk/client-sts";
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

Assumez le rôle IAM.

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });

//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);
```

```
  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK for JavaScript API.

PowerShell

Outils pour PowerShell

Exemple 1 : renvoie un ensemble d'informations d'identification temporaires (clé d'accès, clé secrète et jeton de session) qui peuvent être utilisées pendant une heure pour accéder à AWS des ressources auxquelles l'utilisateur demandeur n'a pas normalement accès. Les informations d'identification renvoyées disposent des autorisations autorisées par la politique d'accès du rôle assumé et par la politique fournie (vous ne pouvez pas utiliser la politique fournie pour accorder des autorisations supérieures à celles définies par la politique d'accès du rôle assumé).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Exemple 2 : renvoie un ensemble d'informations d'identification temporaires, valides pendant une heure, dotées des mêmes autorisations que celles définies dans la politique d'accès du rôle assumé.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Exemple 3 : renvoie un ensemble d'informations d'identification temporaires fournissant le numéro de série et le jeton généré à partir d'un MFA associé aux informations d'identification utilisateur utilisées pour exécuter l'applet de commande.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Exemple 4 : renvoie un ensemble d'informations d'identification temporaires qui ont assumé un rôle défini dans un compte client. Pour chaque rôle que le tiers peut assumer, le compte client doit créer un rôle à l'aide d'un identifiant qui doit être transmis dans le ExternalId paramètre - chaque fois que le rôle est assumé.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Assumez un rôle IAM qui nécessite un jeton MFA et utilisez des informations d'identification temporaires pour répertorier les compartiments Amazon S3 pour le compte.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
```

```
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Pour plus de détails sur l'API, consultez AssumeRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: "create-use-assume-role-scenario"
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
}
```

- Pour plus de détails sur l'API, voir AssumeRolela section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
                async throws -> STSClientTypes.Credentials {
    let input = AssumeRoleInput(
        roleArn: role.arn,
        roleSessionName: sessionName
    )
    do {
        let output = try await stsClient.assumeRole(input: input)

        guard let credentials = output.credentials else {
            throw ServiceHandlerError.authError
        }

        return credentials
    } catch {
        throw error
    }
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section AWS SDK pour la référence de l'API Swift.

Trouver les AWS informations d'identification non utilisées

Pour renforcer la sécurité de votre compte Compte AWS, supprimez les informations d'identification utilisateur IAM (c'est-à-dire les mots de passe et les clés d'accès) qui ne sont pas nécessaires. Par exemple, lorsque des utilisateurs quittent votre organisation ou n'ont plus besoin d'y AWS accéder, recherchez les informations d'identification qu'ils utilisaient et assurez-vous qu'elles ne sont plus opérationnelles. Idéalement, supprimez les informations d'identification qui ne sont plus requises. Il est toujours possible de les recréer ultérieurement, si nécessaire. Vous devez au moins modifier le mot de passe ou désactiver les clés d'accès afin que les anciens utilisateurs ne soient plus en mesure de s'en servir.

La signification du mot inutilisées peut bien sûr varier : cela indique généralement que les informations d'identification n'ont pas été utilisées au cours d'un laps de temps spécifié.

Recherche de mots de passe inutilisés

Vous pouvez utiliser le AWS Management Console pour consulter les informations relatives à l'utilisation des mots de passe par vos utilisateurs. Si vous disposez d'un nombre volumineux d'utilisateurs, vous pouvez utiliser la console pour télécharger un rapport d'informations d'identification contenant des données sur la dernière utilisation du mot de passe de console par chaque utilisateur. Vous pouvez également accéder aux informations à partir de l'API AWS CLI ou de l'API IAM.

Pour rechercher les mots de passe inutilisés (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Au besoin, ajoutez la colonne Console last sign-in (Dernière connexion à la console) à la table des utilisateurs :

   a. Au-dessus de la table à l'extrême droite, choisissez l'icône des paramètres ( ).

   b. Dans Sélectionner les colonnes visibles, sélectionnez Dernière connexion à la console.

c. Choisissez Confirmer pour revenir à la liste des utilisateurs.

4. La colonne Dernière connexion à la console indique la date à laquelle l'utilisateur s'est connecté pour la dernière fois AWS via la console. Ces informations vous permettent de rechercher les utilisateurs avec mots de passe ne s'étant pas connectés depuis une période donnée. La colonne affiche Jamais pour les utilisateurs avec mots de passe ne s'étant jamais connectés. Aucun indique les utilisateurs sans mot de passe. Les mots de passe qui n'ont pas été utilisés récemment peuvent être supprimés.

> **Important**
>
> En raison d'un problème de service, les données de dernière d'utilisation du mot de passe n'incluent pas l'utilisation du mot de passe entre le 3 mai 2018 et le 23 mai 2018 22:50 14:08 PDT (heure du Pacifique). Cela affecte les dates de dernière connexion affichées dans la console IAM et les dates de dernière utilisation du mot de passe dans le rapport d'identification IAM, et renvoyées par l'opération d'API. GetUser Si des utilisateurs se sont connectés au cours de la période concernée, la date de dernière d'utilisation du mot de passe renvoyée est la date de la dernière connexion de l'utilisateur avant le 3 mai 2018. Pour les utilisateurs qui se sont connectés après le 23 mai 2018 14:08 PDT, la date de dernière utilisation du mot de passe renvoyée est exacte.
> Si vous utilisez les informations de dernière utilisation du mot de passe pour identifier les informations d'identification non utilisées à supprimer, par exemple en supprimant des utilisateurs qui ne se AWS sont pas connectés au cours des 90 derniers jours, nous vous recommandons d'ajuster votre fenêtre d'évaluation pour inclure les dates postérieures au 23 mai 2018. Par ailleurs, si vos utilisateurs utilisent des clés d'accès pour accéder AWS par programmation, vous pouvez vous référer aux dernières informations utilisées sur les clés d'accès, car elles sont exactes pour toutes les dates.

Pour rechercher les mots de passe inutilisés en téléchargeant le rapport d'informations d'identification (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Rapport sur les informations d'identification.

3. Choisissez Télécharger le rapport pour télécharger un fichier CSV nommé `status_reports_<date>T<time>.csv`. La cinquième colonne contient la colonne `password_last_used` avec les dates ou l'une des mentions suivantes :

   - N/A : utilisateurs auxquels aucun mot de passe n'est affecté.
   - no_information : utilisateurs n'ayant pas utilisé leur mot de passe depuis le 20 octobre 2014, date à laquelle IAM a commencé le suivi de l'âge des mots de passe.

Pour rechercher des mots de passe inutilisés (AWS CLI)

Exécutez la commande suivante pour rechercher les mots de passe inutilisés :

- `aws iam list-users` retourne une liste d'utilisateurs, avec une valeur `PasswordLastUsed` pour chacun. Si la valeur est manquante, cela signifie que l'utilisateur ne dispose pas de mot de passe ou qu'il ne l'a pas utilisé depuis le 20 octobre 2014, date à laquelle IAM a commencé le suivi de l'âge des mots de passe.

Pour trouver les mots de passe inutilisés (AWS API)

Appelez l'opération suivante pour rechercher les mots de passe inutilisés :

- `ListUsers` retourne une collection d'utilisateurs, avec une valeur `<PasswordLastUsed>` pour chacun. Si la valeur est manquante, cela signifie que l'utilisateur ne dispose pas de mot de passe ou qu'il ne l'a pas utilisé depuis le 20 octobre 2014, date à laquelle IAM a commencé le suivi de l'âge des mots de passe.

Pour plus d'informations sur les commandes à utiliser pour le téléchargement du rapport d'informations d'identification, consultez Obtention de rapports d'informations d'identification (AWS CLI).

Recherche des clés d'accès inutilisées

Vous pouvez utiliser le AWS Management Console pour consulter les informations d'utilisation des clés d'accès pour vos utilisateurs. Si vous disposez d'un nombre volumineux d'utilisateurs, vous pouvez utiliser la console pour télécharger un rapport d'informations d'identification pour connaître la dernière utilisation des clés d'accès par chaque utilisateur. Vous pouvez également accéder aux informations à partir de l'API AWS CLI ou de l'API IAM.

Pour rechercher les clés d'accès inutilisées (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Utilisateurs.
3. Au besoin, ajoutez la colonne Access key last used (Dernière utilisation de clé d'accès) à la table des utilisateurs :
   a. Au-dessus de la table à l'extrême droite, choisissez l'icône des paramètres ( ).
   b. Dans Sélectionner les colonnes visibles, sélectionnez Dernière clé d'accès utilisée.
   c. Choisissez Confirmer pour revenir à la liste des utilisateurs.
4. La colonne Clé d'accès utilisée pour la dernière fois indique le nombre de jours écoulés depuis le dernier accès par AWS programme de l'utilisateur. Ces informations vous permettent de rechercher les utilisateurs avec des clés d'accès n'ayant pas été utilisées depuis une période donnée. La colonne affiche – pour les utilisateurs sans clés d'accès. Les clés d'accès qui n'ont pas été utilisées récemment peuvent être supprimées.

Pour rechercher les clés d'accès inutilisées en téléchargeant le rapport d'informations d'identification (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Rapport sur les informations d'identification.
3. Choisissez Télécharger le rapport pour télécharger un fichier CSV nommé `status_reports_<date>T<time>.csv`. Les colonnes 11 à 13 contiennent la date, la région et les informations de service de la dernière utilisation pour la clé d'accès 1. Les colonnes 16 à 18 contiennent les mêmes informations pour la clé d'accès 2. La valeur est N/A si l'utilisateur ne dispose pas de clé d'accès ou s'il ne l'a pas utilisée depuis le 22 avril 2015, date à laquelle IAM a commencé le suivi de l'âge des clés d'accès.

Pour rechercher les clés d'accès inutilisées (AWS CLI)

Exécutez les commandes suivantes pour rechercher les clés d'accès inutilisées :

- `aws iam list-access-keys` retourne des informations sur les clés d'accès d'un utilisateur, y compris l'`AccessKeyID`.
- `aws iam get-access-key-last-used` utilise un ID de clé d'accès et retourne une sortie qui inclut la `LastUsedDate`, la `Region` dans laquelle la clé d'accès a été utilisée la dernière fois et le `ServiceName` du dernier service demandé. Si la valeur de `LastUsedDate` est manquante, ceci indique que la clé d'accès n'a pas été utilisée depuis le 22 avril 2015, date à laquelle IAM a commencé le suivi de l'âge des clés d'accès.

Pour trouver les clés d'accès non utilisées (AWS API)

Appelez les opérations suivantes pour rechercher les clés d'accès inutilisées :

- `ListAccessKeys` retourne une liste de valeurs `AccessKeyID` pour les clés d'accès associées à l'utilisateur spécifié.
- `GetAccessKeyLastUsed` utilise un ID de clé d'accès et retourne une collection de valeurs. Il s'agit notamment de la `LastUsedDate`, de la `Region` dans laquelle la clé d'accès a été utilisée la dernière fois et du `ServiceName` du dernier service demandé. Si la valeur est manquante, cela signifie que l'utilisateur ne dispose pas de clé d'accès ou qu'il ne l'a pas utilisée depuis le 22 avril 2015, date à laquelle IAM a commencé le suivi de l'âge des clés d'accès.

Pour plus d'informations sur les commandes à utiliser pour le téléchargement du rapport d'informations d'identification, consultez Obtention de rapports d'informations d'identification (AWS CLI).

Obtenir des rapports d'informations d'identification pour votre Compte AWS

Vous pouvez générer et télécharger un rapport sur les informations d'identification qui répertorie tous les utilisateurs de votre compte et le statut de leurs diverses informations d'identification, notamment leurs mots de passe, clés d'accès et dispositifs MFA. Vous pouvez obtenir un rapport d'identification auprès des AWS Management ConsoleAWS SDK et outils de ligne de commande, ou de l'API IAM.

Vous pouvez utiliser les rapports sur les informations d'identification à des fins d'audit et de conformité. Vous pouvez utiliser les rapports pour auditer les effets des exigences en matière de cycle de vie des informations d'identification, comme la mise à jour de mot de passe et de clé d'accès. Vous pouvez fournir les rapports à un auditeur externe ou accorder des autorisations à un auditeur pour qu'il télécharge les rapports directement.

Vous pouvez générer un rapport sur les informations d'identification à quelques heures d'intervalle. Lorsque vous demandez un rapport, IAM vérifie d'abord si un rapport Compte AWS a été généré au cours des quatre dernières heures. Le cas échéant, le rapport le plus récent est téléchargé. Si le rapport le plus récent pour le compte a été généré il y a plus de quatre heures ou qu'il n'y a pas eu d'autres rapport généré précédemment pour le compte, IAM génère et télécharge un nouveau rapport.

Rubriques

- Autorisations nécessaires
- Présentation du format du rapport
- Obtention de rapports d'informations d'identification (console)
- Obtention de rapports d'informations d'identification (AWS CLI)
- Obtenir des rapports d'identification (AWS API)

Autorisations nécessaires

Les autorisations suivantes sont nécessaires pour créer et télécharger des rapports :

- Pour créer un rapport sur les informations d'identification : `iam:GenerateCredentialReport`
- Pour télécharger le rapport : `iam:GetCredentialReport`

Présentation du format du rapport

Les rapports sur les informations d'identification sont formatés comme des fichiers CSV (valeurs séparées par une virgule). Vous pouvez ouvrir les fichiers CSV avec des logiciels de feuilles de calcul courants pour effectuer des analyses ou vous pouvez créer une application qui consomme les fichiers CSV par programmation et effectue des analyses personnalisées.

Le fichier CSV contient les colonnes suivantes :

utilisateur

Nom convivial de l'utilisateur.

arn

Amazon Resource Name (ARN) de l'utilisateur. Pour plus d'informations sur les ARN, consultez ARN IAM.

user_creation_time

Date et heure de création de l'utilisateur, au format de date et d'heure ISO 8601.

password_enabled

Lorsqu'un utilisateur dispose d'un mot de passe, cette valeur est `TRUE`. Sinon, c'est `FALSE` le cas. La valeur de Utilisateur racine d'un compte AWS est toujours`not_supported`.

password_last_used

Date et heure auxquelles le mot de passe de l'utilisateur Utilisateur racine d'un compte AWS ou de l'utilisateur a été utilisé pour la dernière fois pour se connecter à un AWS site Web, au format date-heure ISO 8601. AWS les sites Web qui enregistrent l'heure de dernière connexion d'un utilisateur sont AWS Management Console les forums de AWS discussion et le AWS Marketplace. Quand un mot de passe est utilisé plusieurs fois dans un intervalle de 5 minutes, seule la première utilisation est enregistrée dans ce champ.

- La valeur de ce champ est `no_information` dans les cas suivants :
  - Le mot de passe de l'utilisateur n'a jamais été utilisé.
  - Aucune donnée de connexion n'est associée au mot de passe, comme lorsque le mot de passe de l'utilisateur n'a pas été utilisé depuis le 20 octobre 2014, date de début du suivi de cette information par IAM.
- La valeur de ce champ est `N/A` (non applicable) lorsque l'utilisateur ne dispose pas de mot de passe.

Important

En raison d'un problème de service, les données de dernière d'utilisation du mot de passe n'incluent pas l'utilisation du mot de passe entre le 3 mai 2018 et le 23 mai 2018 22:50 14:08 PDT (heure du Pacifique). Cela affecte les dates de dernière connexion affichées dans la console IAM et les dates de dernière utilisation du mot de passe dans le rapport d'identification IAM, et renvoyées par l'opération d'API. GetUser Si des utilisateurs se sont connectés au cours de la période concernée, la date de dernière d'utilisation du mot de passe renvoyée est la date de la dernière connexion de l'utilisateur avant le 3 mai 2018. Pour les utilisateurs qui se sont connectés après le 23 mai 2018 14:08 PDT, la date de dernière utilisation du mot de passe renvoyée est exacte.
Si vous utilisez les informations de dernière utilisation du mot de passe pour identifier les informations d'identification non utilisées à supprimer, par exemple en supprimant des

utilisateurs qui ne se AWS sont pas connectés au cours des 90 derniers jours, nous vous recommandons d'ajuster votre fenêtre d'évaluation pour inclure les dates postérieures au 23 mai 2018. Par ailleurs, si vos utilisateurs utilisent des clés d'accès pour accéder AWS par programmation, vous pouvez vous référer aux dernières informations utilisées sur les clés d'accès, car elles sont exactes pour toutes les dates.

password_last_changed

Date et heure de la dernière définition du mot de passe de l'utilisateur, au format de date et d'heure ISO 8601. Si l'utilisateur ne dispose pas d'un mot de passe, la valeur de ce champ est `N/A` (non applicable). La valeur de Compte AWS (root) est toujours`not_supported`.

password_next_rotation

Lorsque le compte dispose d'une politique de mot de passe qui nécessite la rotation du mot de passe, ce champ contient la date et l'heure, au format de date et d'heure ISO 8601, auxquelles l'utilisateur doit définir un nouveau mot de passe. La valeur de Compte AWS (root) est toujours`not_supported`.

mfa_active

Lorsqu'un périphérique d'authentification multi-facteur (MFA) a été activé pour l'utilisateur, cette valeur est `TRUE`. Sinon, la valeur est définie comme `FALSE`.

access_key_1_active

Lorsque l'utilisateur dispose d'une clé d'accès et que l'état de celle-ci est `Active`, cette valeur est `TRUE`. Sinon, la valeur est définie comme `FALSE`.

access_key_1_last_rotated

Date et heure, au format de date et d'heure ISO 8601, auxquelles la clé d'accès de l'utilisateur a été créée ou modifiée pour la dernière fois. Si l'utilisateur ne dispose pas d'une clé d'accès active, la valeur de ce champ est `N/A` (non applicable).

access_key_1_last_used_date

Date et heure, au format de date et d'heure ISO 8601, auxquelles la clé d'accès de l'utilisateur a été utilisée pour la dernière fois pour se connecter à une demande d'API AWS . Quand une clé d'accès est utilisée plusieurs fois dans un intervalle de 15 minutes, seule la première utilisation est enregistrée dans ce champ.

La valeur de ce champ est `N/A` (non applicable) dans les cas suivants :

- L'utilisateur ne dispose pas d'une clé d'accès.
- La clé d'accès n'a jamais été utilisée.
- La clé d'accès n'a pas été utilisée depuis le 22 avril 2015, date de début du suivi de cette information par IAM.

access_key_1_last_used_region

Région AWS dans laquelle la clé d'accès a été utilisée pour la dernière fois. Quand une clé d'accès est utilisée plusieurs fois dans un intervalle de 15 minutes, seule la première utilisation est enregistrée dans ce champ.

La valeur de ce champ est `N/A` (non applicable) dans les cas suivants :

- L'utilisateur ne dispose pas d'une clé d'accès.
- La clé d'accès n'a jamais été utilisée.
- La clé d'accès a été utilisée pour la dernière fois avant le 22 avril 2015, date de début du suivi de cette information par IAM.
- Le dernier service utilisé n'est pas spécifique à une région, comme Amazon S3.

access_key_1_last_used_service

Le AWS service auquel vous avez accédé le plus récemment à l'aide de la clé d'accès. La valeur de ce champ utilise l'espace de noms du service (par exemple, `s3` pour Amazon S3 et `ec2` pour Amazon EC2). Quand une clé d'accès est utilisée plusieurs fois dans un intervalle de 15 minutes, seule la première utilisation est enregistrée dans ce champ.

La valeur de ce champ est `N/A` (non applicable) dans les cas suivants :

- L'utilisateur ne dispose pas d'une clé d'accès.
- La clé d'accès n'a jamais été utilisée.
- La clé d'accès a été utilisée pour la dernière fois avant le 22 avril 2015, date de début du suivi de cette information par IAM.

access_key_2_active

Lorsque l'utilisateur dispose d'une seconde clé d'accès et que l'état de celle-ci est `Active`, cette valeur est `TRUE`. Sinon, la valeur est définie comme `FALSE`.

> **Note**
>
> Les utilisateurs peuvent avoir jusqu'à deux clés d'accès, afin de faciliter la rotation en mettant d'abord à jour la clé, puis en supprimant la clé précédente. Pour plus

d'informations sur la mise à jour des clés d'accès, veuillez consulter Mise à jour des clés d'accès.

access_key_2_last_rotated

Date et heure, au format de date et d'heure ISO 8601, de la création ou de la dernière mise à jour de la deuxième clé d'accès de l'utilisateur. Si l'utilisateur ne dispose pas d'une seconde clé d'accès active, la valeur de ce champ est `N/A` (non applicable).

access_key_2_last_used_date

Date et heure, au format date-heure ISO 8601, auxquelles la deuxième clé d'accès de l'utilisateur a été utilisée pour la dernière fois pour signer une AWS demande d'API. Quand une clé d'accès est utilisée plusieurs fois dans un intervalle de 15 minutes, seule la première utilisation est enregistrée dans ce champ.

La valeur de ce champ est `N/A` (non applicable) dans les cas suivants :

- L'utilisateur ne dispose pas d'une seconde clé d'accès.
- La seconde clé d'accès de l'utilisateur n'a jamais été utilisée.
- La seconde clé d'accès de l'utilisateur a été utilisée pour la dernière fois avant le 22 avril 2015, date de début du suivi de cette information par IAM.

access_key_2_last_used_region

Région AWS dans laquelle la seconde clé d'accès de l'utilisateur a été utilisée pour la dernière fois. Quand une clé d'accès est utilisée plusieurs fois dans un intervalle de 15 minutes, seule la première utilisation est enregistrée dans ce champ. La valeur de ce champ est `N/A` (non applicable) dans les cas suivants :

- L'utilisateur ne dispose pas d'une seconde clé d'accès.
- La seconde clé d'accès de l'utilisateur n'a jamais été utilisée.
- La seconde clé d'accès de l'utilisateur a été utilisée pour la dernière fois avant le 22 avril 2015, date de début du suivi de cette information par IAM.
- Le dernier service utilisé n'est pas spécifique à une région, comme Amazon S3.

access_key_2_last_used_service

Le AWS service auquel l'utilisateur a accédé le plus récemment avec la deuxième clé d'accès. La valeur de ce champ utilise l'espace de noms du service (par exemple, `s3` pour Amazon S3 et

`ec2` pour Amazon EC2). Quand une clé d'accès est utilisée plusieurs fois dans un intervalle de 15 minutes, seule la première utilisation est enregistrée dans ce champ. La valeur de ce champ est `N/A` (non applicable) dans les cas suivants :

- L'utilisateur ne dispose pas d'une seconde clé d'accès.
- La seconde clé d'accès de l'utilisateur n'a jamais été utilisée.
- La seconde clé d'accès de l'utilisateur a été utilisée pour la dernière fois avant le 22 avril 2015, date de début du suivi de cette information par IAM.

cert_1_active

Lorsque l'utilisateur dispose d'un certificat de signature X.509 et que l'état de celui-ci est `Active`, cette valeur est `TRUE`. Sinon, la valeur est définie comme `FALSE`.

cert_1_last_rotated

Date et heure, au format de date et d'heure ISO 8601, auxquelles le certificat de signature de l'utilisateur a été créé ou modifié pour la dernière fois. Si l'utilisateur ne dispose pas d'un certificat de signature actif, la valeur de ce champ est `N/A` (non applicable).

cert_2_active

Lorsque l'utilisateur dispose d'un second certificat de signature X.509 et que l'état de celui-ci est `Active`, cette valeur est `TRUE`. Sinon, la valeur est définie comme `FALSE`.

> **Note**
>
> Les utilisateurs peuvent disposer de deux certificats de signature X.509 afin de faciliter la rotation des certificats.

cert_2_last_rotated

Date et heure, au format de date et d'heure ISO 8601, auxquelles le second certificat de signature de l'utilisateur a été créé ou modifié pour la dernière fois. Si l'utilisateur ne dispose pas d'un second certificat de signature actif, la valeur de ce champ est `N/A` (non applicable).

Obtention de rapports d'informations d'identification (console)

Vous pouvez utiliser le AWS Management Console pour télécharger un rapport d'identification sous forme de fichier CSV (valeurs séparées par des virgules).

Pour télécharger un rapport sur les informations d'identification (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Rapport sur les informations d'identification.
3. Choisissez Télécharger le rapport.

Obtention de rapports d'informations d'identification (AWS CLI)

Pour télécharger un rapport sur les informations d'identification (AWS CLI)

1. Générez un rapport sur les informations d'identification. AWS stocke un seul rapport. Si un rapport existe, la génération d'un rapport sur les informations d'identification remplace le rapport précédent. `aws iam generate-credential-report`
2. Affichez le dernier rapport généré : `aws iam get-credential-report`

Obtenir des rapports d'identification (AWS API)

Pour télécharger un rapport sur les informations d'identification (AWS API)

1. Générez un rapport sur les informations d'identification. AWS stocke un seul rapport. Si un rapport existe, la génération d'un rapport sur les informations d'identification remplace le rapport précédent. `GenerateCredentialReport`
2. Affichez le dernier rapport généré : `GetCredentialReport`

Utilisation d'IAM avec CodeCommit : informations d'identification Git, clés SSH et AWS clés d'accès

CodeCommit est un service de contrôle de version géré qui héberge des référentiels Git privés dans le AWS cloud. Pour l'utiliser CodeCommit, vous devez configurer votre client Git pour qu'il communique avec CodeCommit les référentiels. Dans le cadre de cette configuration, vous fournissez des informations d'identification IAM qui CodeCommit peuvent être utilisées pour vous authentifier. IAM prend en charge CodeCommit trois types d'informations d'identification :

- Informations d'identification Git, paire nom d'utilisateur et mot de passe générée par IAM que vous pouvez utiliser pour communiquer avec les CodeCommit référentiels via HTTPS.

- Les clés SSH sont une paire de clés publique-privée générée localement que vous pouvez associer à votre utilisateur IAM pour communiquer avec les CodeCommit référentiels via SSH.
- AWS clés d'accès, que vous pouvez utiliser avec l'assistant d'identification inclus dans le AWS CLI pour communiquer avec les CodeCommit référentiels via HTTPS.

> **Note**
>
> Vous ne pouvez pas utiliser les clés SSH ni les informations d'identification Git pour accéder aux référentiels dans un autre compte AWS . Pour savoir comment configurer l'accès aux CodeCommit référentiels pour les utilisateurs et les groupes IAM d'un autre utilisateur Compte AWS, voir Configurer l'accès entre comptes à un AWS CodeCommit référentiel à l'aide de rôles dans le Guide de l'AWS CodeCommit utilisateur.

Pour plus d'informations sur chaque option, consultez les sections suivantes.

Utiliser les informations d'identification Git et HTTPS avec CodeCommit (recommandé)

Avec les informations d'identification Git, vous générez une paire mot de passe-nom utilisateur statique pour votre utilisateur IAM, puis utilisez ces informations d'identification pour les connexions HTTPS. Vous pouvez également utiliser ces informations d'identification avec n'importe quel outil tiers ou environnement de développement intégré (IDE) prenant en charge les informations d'identification Git statiques.

Etant donné que ces informations d'identification sont universelles pour tous les systèmes d'exploitation pris en charge et compatibles avec la plupart des systèmes de gestion des informations d'identification, environnements de développement et autres outils de développement logiciel, c'est la méthode recommandée. Vous pouvez réinitialiser le mot de passe pour les informations d'identification Git à tout moment. Vous pouvez également désactiver les informations d'identification ou les supprimer si vous n'en avez plus besoin.

> **Note**
>
> Pour les informations d'identification Git, vous ne pouvez pas choisir vous-même votre nom d'utilisateur et votre mot de passe. IAM génère ces informations d'identification pour vous aider à garantir qu'elles répondent aux normes de sécurité AWS et qu'elles sécurisent les référentiels dans. CodeCommit Vous pouvez télécharger les informations d'identification

une seule fois, au moment elles sont générées. Veillez à enregistrer les informations d'identification dans un emplacement sûr. Si nécessaire, vous pouvez réinitialiser le mot de passe à tout moment, mais cela invalidera les connexions configurées avec l'ancien mot de passe. Vous devez reconfigurer les connexions pour qu'elles utilisent le nouveau mot de passe avant de vous connecter.

Pour plus d'informations, consultez les rubriques suivantes :

- Pour créer un utilisateur IAM, veuillez consulter Création d'un utilisateur IAM dans votre Compte AWS.
- Pour générer et utiliser des informations d'identification Git avec CodeCommit, consultez la section Pour les utilisateurs HTTPS utilisant des informations d'identification Git dans le guide de AWS CodeCommit l'utilisateur.

> **Note**
>
> Le fait de modifier le nom d'un utilisateur IAM après avoir généré des informations d'identification Git ne modifie pas le nom utilisateur des informations d'identification Git. Le nom utilisateur et le mot de passe restent identiques et sont toujours valides.

Pour mettre à jour les informations d'identification spécifiques au service

1. Créez un second ensemble d'informations d'identification spécifiques au service en plus de l'ensemble actuellement en cours d'utilisation.
2. Mettez à jour toutes vos applications afin qu'elles utilisent le nouvel ensemble d'informations d'identification et vérifiez que les applications fonctionnent.
3. Modifiez l'état des informations d'identification d'origine et utilisez Inactif.
4. Vérifiez que toutes vos applications fonctionnent toujours.
5. Supprimer les informations d'identification spécifiques au service inactives.

Utilisez des clés SSH et SSH avec CodeCommit

Avec les connexions SSH, vous créez des fichiers de clés publiques et privées sur votre machine locale que Git CodeCommit utilise pour l'authentification SSH. Vous associez la clé publique à votre

utilisateur IAM et stockez la clé privée sur votre ordinateur local. Pour plus d'informations, consultez les rubriques suivantes :

- Pour créer un utilisateur IAM, veuillez consulter Création d'un utilisateur IAM dans votre Compte AWS.
- Pour créer une clé publique SSH et l'associer à un utilisateur IAM, consultez Pour les connexions SSH sous Linux, macOS ou Unix ou Pour des connexions SSH sous Windows dans le Guide de l'utilisateur AWS CodeCommit .

> **Note**
>
> La clé publique doit être codée au format ssh-rsa ou PEM. Le longueur binaire minimale de la clé publique est de 2048 bits et la longueur maximale de 16 384 bits. Cette longueur est distincte de la taille du fichier que vous chargez. Par exemple, vous pouvez générer une clé de 2048 bits et le fichier PEM résultant a une longueur de 1679 octets. Si vous fournissez votre clé publique dans un autre format ou une autre taille, un message d'erreur vous indiquera que le format de clé n'est pas valide.

Utilisez HTTPS avec l'assistant AWS CLI d'identification et CodeCommit

Comme alternative aux connexions HTTPS avec des informations d'identification Git, vous pouvez autoriser Git à utiliser une version signée cryptographiquement de vos informations d'identification utilisateur IAM ou de votre rôle d'instance Amazon EC2 chaque fois que Git doit s'authentifier pour interagir AWS avec des référentiels. CodeCommit Il s'agit de la seule méthode de connexion pour CodeCommit les référentiels qui ne nécessite pas d'utilisateur IAM. C'est également la seule méthode qui fonctionne avec un accès fédéré et des informations d'identification temporaires. Pour plus d'informations, consultez les rubriques suivantes :

- Pour en savoir plus sur l'accès fédéré, consultez Fournisseurs d'identité et fédération et Octroyer l'accès à des utilisateurs authentifiés en externe (fédération d'identité).
- Pour en savoir plus sur les informations d'identification temporaires, consultez Informations d'identification de sécurité temporaires dans IAM la section Accès temporaire aux CodeCommit référentiels.

L'assistant AWS CLI d'identification n'est pas compatible avec les autres systèmes d'assistance aux identifiants, tels que Keychain Access ou Windows Credential Management. Il existe d'autres

considérations relatives à la configuration lorsque vous configurez des connexions HTTPS à l'aide de l'assistant d'informations d'identification. Pour plus d'informations, voir Pour les connexions HTTPS sous Linux, macOS ou Unix avec l'assistant AWS CLI d'identification ou Connexions HTTPS sous Windows avec l'assistant AWS CLI d'identification dans le guide de l'utilisateur.AWS CodeCommit

Utilisation d'IAM avec Amazon Keyspaces (pour Apache Cassandra)

Amazon Keyspaces (pour Apache Cassandra) est un service de base de données compatible avec Apache Cassandra, évolutif, hautement disponible et géré. Vous pouvez accéder à Amazon Keyspaces via le ou par AWS Management Console programme. Pour accéder à Amazon Keyspaces de manière programmatique avec des informations d'identification spécifiques au service, vous pouvez utiliser `cqlsh` ou des pilotes Cassandra open source. Les informations d'identification spécifiques à un service comprennent un nom d'utilisateur et un mot de passe similaires à ceux que Cassandra utilise pour l'authentification et la gestion des accès. Vous pouvez avoir un maximum de deux ensembles d'informations d’identification spécifiques au service pour chaque service pris en charge par utilisateur.

Pour accéder à Amazon Keyspaces par programmation à l'aide de clés d' AWS accès, vous pouvez utiliser le AWS SDK, le AWS Command Line Interface (AWS CLI) ou les pilotes Cassandra open source avec le plugin SigV4. Pour en savoir plus, consultez la rubrique Connexion programmatique à Amazon Keyspaces dans le Guide du développeur Amazon Keyspaces (pour Apache Cassandra).

> **Note**
>
> Si vous envisagez d'interagir avec Amazon Keyspaces uniquement via la console, vous n'avez pas besoin de générer des informations d'identification spécifiques au service. Pour de plus amples informations, veuillez consulter la rubrique Accès à Amazon Keyspaces à l'aide de la console dans le Guide du développeur Amazon Keyspaces (pour Apache Cassandra).

Pour de plus amples informations sur les autorisations requises pour accéder à Amazon Keyspaces, veuillez consulter Exemples de politiques basées sur l'identité Amazon Keyspaces (for Apache Cassandra) dans le Manuel du développeur Amazon Keyspaces (for Apache Cassandra).

Génération d'informations d'identification Amazon Keyspaces (console)

Vous pouvez utiliser le AWS Management Console pour générer des informations d'identification Amazon Keyspaces (pour Apache Cassandra) pour vos utilisateurs IAM.

Pour générer des informations d'identification spécifiques au service Amazon Keyspaces (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Users (Utilisateurs), puis choisissez le nom de l'utilisateur ayant besoin d'informations d'identification.
3. Dans l'onglet Security Credentials (informations d'identification de sécurité) sous Credentials for Amazon Keyspaces (informations d'identification pour Amazon Keyspaces) (pour Apache Cassandra) (MCS) sélectionnez Generate credentials (générer des informations d'identification).
4. Vos informations d'identification spécifiques au service sont disponibles. C'est la seule fois que vous pouvez afficher ou télécharger le mot de passe. Vous ne pourrez pas la récupérer plus tard. Cependant, vous pouvez réinitialiser votre mot de passe à tout moment. Enregistrez cet utilisateur et ce mot de passe dans un emplacement sécurisé, car vous en aurez besoin plus tard.

Génération d'informations d'identification Amazon Keyspaces (AWS CLI)

Vous pouvez utiliser le AWS CLI pour générer des informations d'identification Amazon Keyspaces (pour Apache Cassandra) pour vos utilisateurs IAM.

Pour générer des informations d'identification spécifiques au service Amazon Keyspaces (AWS CLI)

- Utilisez la commande suivante :
  - était un objectif create-service-specific-credential

Génération d'informations d'identification Amazon Keyspaces (API)AWS

Vous pouvez utiliser l' AWS API pour générer des informations d'identification Amazon Keyspaces (pour Apache Cassandra) pour vos utilisateurs IAM.

Pour générer des informations d'identification (API) spécifiques au service Amazon Keyspaces AWS

- Complétez l'opération suivante :
  - CreateServiceSpecificCredential

Gestion des certificats de serveur dans IAM

Pour activer les connexions HTTPS à votre site Web ou à votre application AWS, vous avez besoin d'un certificat de serveur SSL/TLS. Pour les certificats dans une région prise en charge par AWS Certificate Manager (ACM), nous vous recommandons d'utiliser ACM pour allouer, gérer et déployer vos certificats de serveur. Dans les régions non prises en charge, vous devez utiliser IAM en tant que gestionnaire de certificats. Pour savoir quelles régions sont prises en charge par ACM, consultez Points de terminaison et quotas AWS Certificate Manager (français non garanti) dans Références générales AWS.

ACM est l'outil préféré pour mettre en service, gérer et déployer vos certificats de serveur. Avec ACM, vous pouvez demander un certificat ou déployer un certificat ACM existant ou un certificat externe sur les AWS ressources. Les certificats fournis par ACM sont gratuits et renouvelés automatiquement. Dans une région prise en charge, vous pouvez utiliser ACM pour gérer des certificats de serveur depuis la console ou par programmation. Pour plus d'informations sur l'utilisation d'ACM, consultez le Guide de l'utilisateur AWS Certificate Manager . Pour plus d'informations sur la demande d'un certificat ACM, consultez Demander un certificat public ou Demander in certificat privé dans le Guide de l'utilisateur AWS Certificate Manager . Pour plus d'informations sur l'importation de certificats tiers dans ACM, consultez Importation de certificats dans le Guide de l’utilisateur AWS Certificate Manager .

Utilisez IAM comme gestionnaire de certificats uniquement lorsque vous devez prendre en charge des connexions HTTPS dans une région non prise en charge par ACM. IAM chiffre en toute sécurité vos clés privées et stocke la version chiffrée dans le magasin de certificats SSL IAM. IAM prend en charge le déploiement de certificats de serveur dans toutes les régions, mais vous devez obtenir votre certificat auprès d'un fournisseur externe pour pouvoir l'utiliser avec AWS. Vous ne pouvez pas télécharger de certificat ACM dans IAM. De plus, vous ne pouvez pas gérer vos certificats depuis la console IAM.

Pour plus d'informations sur le chargement de certificats tiers dans IAM, consultez les rubriques suivantes.

Table des matières

- Téléchargement d'un certificat de serveur (AWS API)
- Récupération d'un certificat de serveur (AWS API)
- Liste des certificats de serveur (AWS API)
- Ajout et suppression de balises pour les certificats de serveur (API AWS)

- Modification du nom d'un certificat de serveur ou mise à jour de son chemin (AWS API)
- Supprimer un certificat de serveur (AWS API)
- Résolution des problèmes

Téléchargement d'un certificat de serveur (AWS API)

Pour télécharger un certificat de serveur dans IAM, vous devez fournir le certificat et sa clé privée correspondante. Si le certificat n'est pas auto-signé, vous devez également fournir une chaîne de certificats. (Vous n'avez pas besoin d'une chaîne de certificats lorsque vous chargez un certificat auto-signé.) Avant de charger un certificat, vérifiez que vous avez tous ces éléments et qu'ils répondent aux critères suivants :

- Le certificat doit être valide au moment du chargement. Vous ne pouvez pas charger un certificat avant le début de sa période de validité (date `NotBefore` du certificat) ou après son expiration (date `NotAfter` du certificat).
- La clé privée doit être non chiffrée. Vous ne pouvez pas charger une clé privée qui est protégée par un mot de passe ou une phrase passe. Pour obtenir de l'aide pour déchiffrer une clé privée chiffrée, consultez Résolution des problèmes.
- Le certificat, la clé privée et la chaîne de certificats doivent être codés PEM. Pour obtenir de l'aide pour convertir ces éléments au format PEM, consultez Résolution des problèmes.

Pour utiliser l'API IAM afin de télécharger un certificat, envoyez une UploadServerCertificatedemande. L'exemple suivant montre comment procéder avec l'AWS Command Line Interface (AWS CLI). Dans cet exemple il est supposé que :

- Le certificat codé en PEM est stocké dans un fichier nommé `Certificate.pem`.
- La chaîne de certificats codée en PEM est stockée dans un fichier nommé `CertificateChain.pem`.
- La clé privée non chiffrée, codée en PEM est stockée dans un fichier nommé `PrivateKey.pem`.
- (Facultatif) Vous souhaitez baliser le certificat de serveur avec une paire clé-valeur. Par exemple, vous pouvez ajouter la clé de balise `Department` et la valeur de balise `Engineering` pour vous aider à identifier et organiser vos certificats.

Pour utiliser l'exemple de commande suivant, remplacez ces noms de fichier par les vôtres. *`ExampleCertificate`*Remplacez-le par un nom pour le certificat que vous avez téléchargé.

Si vous souhaitez étiqueter le certificat, remplacez la paire clé-valeur *ExampleKey*and *ExampleValue*tag par vos propres valeurs. Tapez la commande sur une seule ligne continue. L'exemple suivant inclut des sauts de ligne et des espaces supplémentaires pour en faciliter la lecture.

```
aws iam upload-server-certificate --server-certificate-name ExampleCertificate
                                  --certificate-body file://Certificate.pem
                                  --certificate-chain file://CertificateChain.pem
                                  --private-key file://PrivateKey.pem
                                  --tags '{"Key": "ExampleKey", "Value":
 "ExampleValue"}'
```

Lorsque la commande précédente aboutit, elle renvoie des métadonnées relatives au certificat chargé, y compris son Amazon Resource Name (ARN), son nom convivial, son identifiant (ID), sa date d'expiration, ses balises, etc.

> **Note**
>
> Si vous téléchargez un certificat de serveur à utiliser avec Amazon CloudFront, vous devez spécifier un chemin à l'aide de l'`--path`option. Le chemin doit commencer par `/cloudfront` et doit inclure une barre oblique de fin (par exemple, `/cloudfront/test/`).

AWS Tools for Windows PowerShell Pour télécharger un certificat, utilisez ServerCertificatePublish-IAM.

Récupération d'un certificat de serveur (AWS API)

Pour utiliser l'API IAM afin de récupérer un certificat, envoyez une GetServerCertificatedemande. L'exemple suivant montre comment procéder avec l'interface AWS CLI. Remplacez *ExampleCertificate*par le nom du certificat à récupérer.

```
aws iam get-server-certificate --server-certificate-name ExampleCertificate
```

Lorsque la commande précédente aboutit, elle renvoie le certificat, la chaîne de certificats (si une chaîne de certificats a été chargée) et des métadonnées relatives au certificat.

> **Note**
>
> Vous ne pouvez pas télécharger ou récupérer une clé privée depuis IAM après l'avoir téléchargée.

AWS Tools for Windows PowerShell Pour récupérer un certificat, utilisez Get-IAM ServerCertificate.

Liste des certificats de serveur (AWS API)

Pour utiliser l'API IAM afin de répertorier les certificats de serveur que vous avez téléchargés, envoyez une ListServerCertificatesdemande. L'exemple suivant montre comment procéder avec l'interface AWS CLI.

```
aws iam list-server-certificates
```

Lorsque la commande précédente aboutit, elle renvoie une liste qui contient les métadonnées relatives à chaque certificat.

Pour répertorier les certificats AWS Tools for Windows PowerShell de serveur que vous avez téléchargés, utilisez Get-IAM ServerCertificates.

Ajout et suppression de balises pour les certificats de serveur (API AWS)

Vous pouvez attacher des balises à vos ressources IAM afin d'organiser et de contrôler leur accès. Pour utiliser l'API IAM afin de baliser un certificat de serveur existant, envoyez une TagServerCertificatedemande. L'exemple suivant montre comment procéder avec l'interface AWS CLI.

```
aws iam tag-server-certificate --server-certificate-name ExampleCertificate
                                --tags '{"Key": "ExampleKey", "Value":
 "ExampleValue"}'
```

Lorsque la commande précédente aboutit, aucune sortie n'est renvoyée.

Pour utiliser l'API IAM afin de supprimer le balisage d'un certificat de serveur, envoyez une UntagServerCertificatedemande. L'exemple suivant montre comment procéder avec l'interface AWS CLI.

```
aws iam untag-server-certificate --server-certificate-name ExampleCertificate
                                 --tag-keys ExampleKeyName
```

Lorsque la commande précédente aboutit, aucune sortie n'est renvoyée.

Modification du nom d'un certificat de serveur ou mise à jour de son chemin (AWS API)

Pour utiliser l'API IAM afin de renommer un certificat de serveur ou de mettre à jour son chemin, envoyez une UpdateServerCertificatedemande. L'exemple suivant montre comment procéder avec l'interface AWS CLI.

Pour utiliser la commande suivante, remplacez les anciens et nouveaux noms de certificat et le chemin d'accès du certificat, et tapez la commande sur une seule ligne continue. L'exemple suivant inclut des sauts de ligne et des espaces supplémentaires pour en faciliter la lecture.

```
aws iam update-server-certificate --server-certificate-name ExampleCertificate
                                  --new-server-certificate-name CloudFrontCertificate
                                  --new-path /cloudfront/
```

Lorsque la commande précédente aboutit, elle ne renvoie pas de sortie.

Pour renommer un certificat AWS Tools for Windows PowerShell de serveur ou mettre à jour son chemin, utilisez ServerCertificateUpdate-IAM.

Supprimer un certificat de serveur (AWS API)

Pour utiliser l'API IAM afin de supprimer un certificat de serveur, envoyez une DeleteServerCertificatedemande. L'exemple suivant montre comment procéder avec l'interface AWS CLI.

Pour utiliser l'exemple de commande suivant, *ExampleCertificate*remplacez-le par le nom du certificat à supprimer.

```
aws iam delete-server-certificate --server-certificate-name ExampleCertificate
```

Lorsque la commande précédente aboutit, elle ne renvoie pas de sortie.

Pour supprimer un certificat AWS Tools for Windows PowerShell de serveur, utilisez ServerCertificateRemove-IAM.

Résolution des problèmes

Pour pouvoir charger un certificat vers IAM, vous devez vous assurer que le certificat, la clé privée et la chaîne de certificats sont tous codés PEM. Vous devez également vous assurer que la clé privée est non chiffrée. Voir les exemples suivantes.

Example Exemple de certificat codé PEM

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Example Clé privée non chiffrée codée PEM

```
-----BEGIN RSA PRIVATE KEY-----
Base64-encoded private key
-----END RSA PRIVATE KEY-----
```

Example Chaîne de certificats codée PEM

Une chaîne de certificats contient un ou plusieurs certificats. Vous pouvez utiliser un éditeur de texte, la commande copy sous Windows ou la commande cat Linux pour concaténer vos fichiers de certificats dans une chaîne. Lorsque vous incluez plusieurs certificats, chacun d'entre eux doit approuver le certificat précédent. Vous pouvez effectuer cette opération en concaténant les certificats, y compris le certificat d'autorité de certification racine en dernier.

L'exemple suivant contient trois certificats, mais votre chaîne de certificats peut en contenir plus ou moins.

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Si ces éléments ne sont pas au bon format pour le téléchargement vers IAM, vous pouvez utiliser OpenSSL pour les convertir au format approprié.

Pour convertir un certificat ou une chaîne de certificats DER en PEM

Utilisez la commande OpenSSL x509, comme dans l'exemple suivant. Dans la commande suivante, remplacez *Certificate.der* par le nom du fichier qui contient votre certificat codé DER. Remplacez *Certificate.pem* par le nom préféré du fichier de sortie devant contenir le certificat codé PEM.

```
openssl x509 -inform DER -in Certificate.der -outform PEM -out Certificate.pem
```

Pour convertir une clé privée de DER en PEM

Utilisez la commande OpenSSL rsa, comme dans l'exemple suivant. Dans la commande suivante, remplacez *PrivateKey.der* par le nom du fichier qui contient votre clé privée codée DER. Remplacez *PrivateKey.pem* par le nom préféré du fichier de sortie devant contenir la clé privée codée PEM.

```
openssl rsa -inform DER -in PrivateKey.der -outform PEM -out PrivateKey.pem
```

Pour déchiffrer une clé privée chiffrée (supprimer le mot de passe ou la phrase de passe)

Utilisez la commande OpenSSL rsa, comme dans l'exemple suivant. Pour utiliser la commande suivante, remplacez *EncryptedPrivateKey.pem* par le nom du fichier qui contient votre clé privée chiffrée. Remplacez *PrivateKey.pem* par le nom préféré du fichier de sortie devant contenir la clé privée non chiffrée codée PEM.

```
openssl rsa -in EncryptedPrivateKey.pem -out PrivateKey.pem
```

Pour convertir un ensemble de certificats de PKCS#12 (PFX) en PEM

Utilisez la commande OpenSSL pkcs12, comme dans l'exemple suivant. Dans la commande suivante, remplacez *CertificateBundle.p12* par le nom du fichier qui contient votre ensemble de certificats codé PKCS#12. Remplacez *CertificateBundle.pem* par le nom préféré du fichier de sortie devant contenir l'ensemble de certificats codé PEM.

```
openssl pkcs12 -in CertificateBundle.p12 -out CertificateBundle.pem -nodes
```

Pour convertir un ensemble de certificats de PKCS#7 en PEM

Utilisez la commande OpenSSL pkcs7, comme dans l'exemple suivant. Dans la commande suivante, remplacez *CertificateBundle.p7b* par le nom du fichier qui contient votre ensemble de certificats codé PKCS#7. Remplacez *CertificateBundle.pem* par le nom préféré du fichier de sortie devant contenir l'ensemble de certificats codé PEM.

```
openssl pkcs7 -in CertificateBundle.p7b -print_certs -out CertificateBundle.pem
```

Groupes d'utilisateurs IAM

Un groupe d'utilisateurs IAM est un ensemble d'utilisateurs IAM. Les groupes d'utilisateurs vous permettent de spécifier des autorisations pour plusieurs utilisateurs, ce qui permet de gérer plus facilement les autorisations pour ces utilisateurs. Par exemple, vous pouvez avoir un groupe d'utilisateurs appelé Admins et accorder à ce groupe d'utilisateurs des autorisations d'administrateur typiques. Tous les utilisateurs de ce groupe d'utilisateurs reçoivent automatiquement les autorisations de groupe Admins. Si un nouvel utilisateur rejoint votre organisation et a besoin de privilèges d'administrateur, vous pouvez lui attribuer les autorisations appropriées en l'ajoutant au groupe d'utilisateurs Admins. Si une personne change de poste dans votre organisation, au lieu de modifier les autorisations de cet utilisateur, vous pouvez retirer celui-ci de l'ancien groupe d'utilisateurs et l'ajouter aux nouveaux groupes d'utilisateurs appropriés.

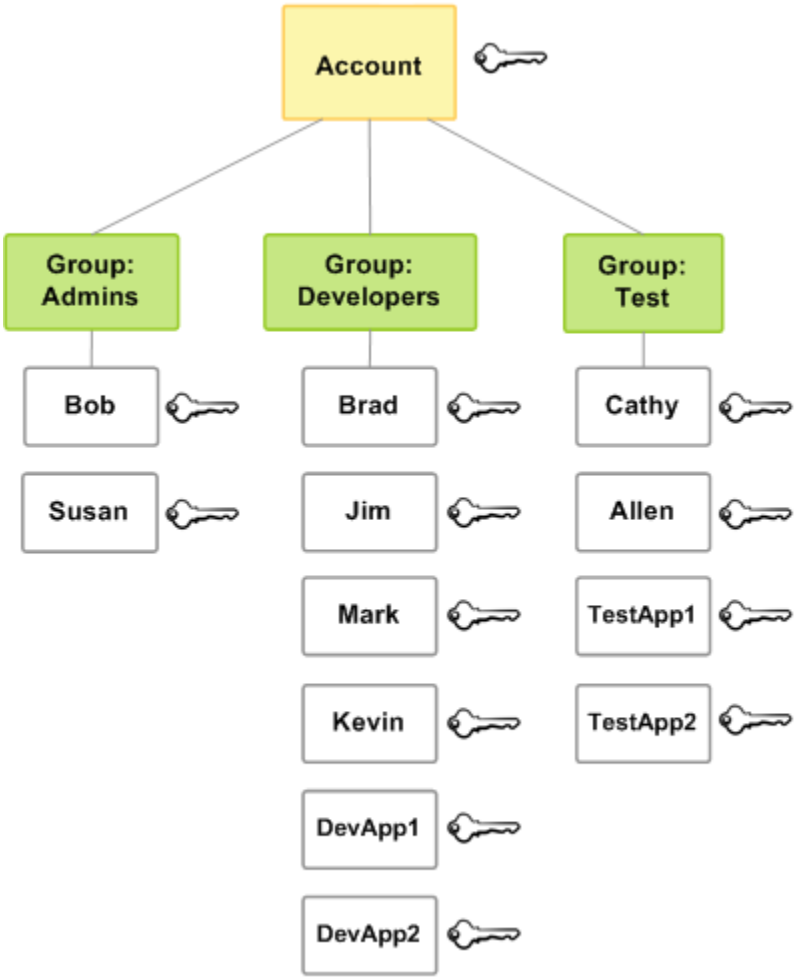
Vous pouvez attacher une politique basée sur l'identité à un groupe d'utilisateurs afin que tous les utilisateurs de ce groupe reçoivent les autorisations de la politique. Vous ne pouvez pas identifier un groupe d'utilisateurs en tant que `Principal` dans une politique (telle qu'une politique basée sur les ressources) car les groupes se rapportent aux autorisations, et non à l'authentification, et les principaux sont des entités IAM authentifiées. Pour de plus amples informations sur les types de politiques, veuillez consulter Politiques basées sur l'identité et Politiques basées sur une ressource.

Voici quelques caractéristiques importantes des groupes d'utilisateurs :

- Un groupe d'utilisateurs peut contenir de nombreux utilisateurs et un utilisateur peut appartenir à plusieurs groupes d'utilisateurs.
- Les groupes d'utilisateurs ne peuvent pas être imbriqués ; ils ne peuvent contenir que des utilisateurs, pas d'autres groupes d'utilisateurs.

- Il n'existe pas de groupe d'utilisateurs par défaut qui inclut automatiquement tous les utilisateurs de l' Compte AWS. Si vous souhaitez disposer d'un groupe d'utilisateurs de ce type, vous devez le créer et lui affecter chaque nouvel utilisateur.
- Le nombre et la taille des ressources IAM dans un Compte AWS, tels que le nombre de groupes et le nombre de groupes dont un utilisateur peut être membre, sont limités. Pour plus d'informations, consultez IAM et quotas AWS STS.

Le schéma suivant illustre un exemple simple de petite entreprise. Le propriétaire de l'entreprise crée un groupe d'utilisateurs `Admins` pour que des utilisateurs créent et gèrent d'autres utilisateurs au fur et à mesure que l'entreprise se développe. Le groupe d'utilisateurs `Admins` crée un groupe d'utilisateurs `Developers` et un groupe d'utilisateurs `Test`. Chacun de ces groupes d'utilisateurs est composé d'utilisateurs (humains et applications) qui interagissent avec AWS (Jim, Brad, DevApp 1, etc.). Chaque utilisateur dispose d'un ensemble individuel d'informations d'identification de sécurité. Dans cet exemple, chaque utilisateur appartient à un seul groupe d'utilisateurs. Cependant, les utilisateurs peuvent appartenir à plusieurs groupes d'utilisateurs.



Création de groupes d'utilisateurs IAM

> **Note**
>
> En tant que bonne pratique, nous vous recommandons de demander aux utilisateurs humains d'utiliser la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires. Si vous suivez les bonnes pratiques, vous ne gérez pas les utilisateurs et les groupes IAM. Au lieu de cela, vos utilisateurs et vos groupes sont gérés en dehors de l'extérieur AWS et peuvent accéder aux AWS ressources en tant qu'identité fédérée. Une identité fédérée est un utilisateur de l'annuaire des utilisateurs de votre entreprise, un fournisseur d'identité Web, le AWS Directory Service, le répertoire Identity Center ou tout utilisateur qui accède AWS aux services à l'aide des informations d'identification fournies par le biais d'une source d'identité. Les identités fédérées utilisent les groupes définis par leur fournisseur d'identité. Si vous en utilisez AWS IAM Identity Center,

consultez la section Gérer les identités dans IAM Identity Center dans le Guide de l'AWS IAM Identity Center utilisateur pour plus d'informations sur la création d'utilisateurs et de groupes dans IAM Identity Center.

Pour configurer un groupe d'utilisateurs, vous devez le créer. Ensuite, attribuez au groupe des autorisations en fonction du type de tâche que les utilisateurs du groupe seront amenés à effectuer. Enfin, ajoutez les utilisateurs au groupe.

Pour plus d'informations sur les autorisations requises pour créer un groupe d'utilisateurs, veuillez consulter Autorisations requises pour accéder aux autres ressources IAM.

Pour créer un groupe IAM et attacher des politiques (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), puis Create group (Créer un groupe).
3. Pour User group name (Nom du groupe d'utilisateurs), saisissez le nom du groupe.

   > **Note**
   >
   > Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d'informations, consultez IAM et quotas AWS STS. Les noms de groupe peuvent combiner jusqu'à 128 lettres, chiffres et caractères suivants : plus (+), égal (=), virgule (,), point (.), arobase (@), trait de soulignement (_) et tiret (-). Les noms doivent être uniques dans un compte. Ils ne sont pas sensibles à la casse. Par exemple, vous ne pouvez pas créer deux groupes nommés **`ADMINS`** et **`admins`**.

4. Dans la liste des utilisateurs, cochez la case en regard de chacun des utilisateurs à ajouter au groupe.
5. Dans la liste de politiques, activez la case à cocher en regard de chaque politique devant s'appliquer à tous les membres du groupe.
6. Choisissez Créer un groupe.

Pour créer des groupes d'utilisateurs IAM (AWS CLI ou AWS API)

Utilisez l'une des options suivantes :

- AWS CLI : aws iam create-group
- AWS API : CreateGroup

Gestion des groupes IAM

Amazon Web Services propose plusieurs outils pour gérer les groupes d'utilisateurs IAM. Pour plus d'informations sur les autorisations dont vous devez disposer pour ajouter ou supprimer des utilisateurs dans un groupe d'utilisateurs, veuillez consulter Autorisations requises pour accéder aux autres ressources IAM.

Rubriques

- Liste de groupes IAM
- Ajout et suppression d'utilisateurs dans un groupe IAM
- Attacher une politique à un groupe IAM
- Affectation d'un nouveau nom à un groupe IAM
- Suppression d'un groupe d'utilisateurs IAM

Liste de groupes IAM

Vous pouvez répertorier tous les groupes d'utilisateurs de votre compte, les utilisateurs d'un groupe d'utilisateurs ou les groupes d'utilisateurs auxquels appartient un utilisateur. Si vous utilisez l' AWS API AWS CLI or, vous pouvez répertorier tous les groupes d'utilisateurs avec un préfixe de chemin particulier.

Pour répertorier tous les groupes d'utilisateurs de votre compte

Effectuez l’une des actions suivantes :

- AWS Management Console : dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs).
- AWS CLI : aws iam list-groups
- AWS API : ListGroups

Pour répertorier les utilisateurs d'un groupe d'utilisateurs spécifique

Effectuez l'une des actions suivantes :

- AWS Management Console : dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), choisissez le nom du groupe, puis sélectionnez l'onglet Users (Utilisateurs).
- AWS CLI : aws iam get-group
- AWS API : GetGroup

Pour répertorier tous les groupes d'utilisateurs auxquels appartient un utilisateur

Effectuez l'une des actions suivantes :

- AWS Management Console : dans le panneau de navigation, sélectionnez Utilisateurs, choisissez le nom d'utilisateur, puis sélectionnez l'onglet Groupes.
- AWS CLI: cime list-groups-for-user
- AWS API : ListGroupsForUser

Ajout et suppression d'utilisateurs dans un groupe IAM

Utilisez les groupes d'utilisateurs pour appliquer les mêmes politiques d'autorisations sur plusieurs utilisateurs à la fois. Vous pouvez ensuite ajouter des utilisateurs à un groupe d'utilisateurs IAM ou en supprimer. Cet utile à mesure que des utilisateurs arrivent dans votre organisation et la quittent.

Afficher l'accès à la politique

Avant de modifier les autorisations d'une politique, vous devez passer en revue ses activités récentes au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Ajouter un utilisateur à un groupe d'utilisateurs ou en supprimer (console)

Vous pouvez utiliser le AWS Management Console pour ajouter ou supprimer un utilisateur d'un groupe d'utilisateurs.

Pour ajouter un utilisateur à un groupe d'utilisateurs IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), puis choisissez le nom du groupe.
3. Sélectionnez l'onglet Users (Utilisateurs), puis Add users (Ajouter des utilisateurs). Cochez la case à côté de l'utilisateur que vous souhaitez ajouter.
4. Sélectionnez Add users (Ajouter des utilisateurs).

Pour supprimer un utilisateur d'un groupe IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), puis choisissez le nom du groupe.
3. Sélectionnez l'onglet Utilisateurs. Cochez la case en regard de l'utilisateur à supprimer, puis choisissez Remove users (Supprimer des utilisateurs).

Ajouter un utilisateur à un groupe d'utilisateurs ou en supprimer (AWS CLI)

Vous pouvez utiliser le AWS CLI pour ajouter ou supprimer un utilisateur d'un groupe d'utilisateurs.

Pour ajouter un utilisateur à un groupe IAM (AWS CLI)

- Utilisez la commande suivante :
  - était un objectif add-user-to-group

Pour supprimer un utilisateur d'un groupe IAM (AWS CLI)

- Utilisez la commande suivante :
  - était un objectif remove-user-from-group

Ajouter ou supprimer un utilisateur dans un groupe d'utilisateurs (AWS API)

Vous pouvez utiliser l' AWS API pour ajouter ou supprimer un utilisateur dans un groupe d'utilisateurs.

Pour ajouter un utilisateur à un groupe IAM (AWS API)

- Complétez l'opération suivante :
  - AddUserToGroup

Pour supprimer un utilisateur d'un groupe d'utilisateurs IAM (AWS API)

- Complétez l'opération suivante :
  - RemoveUserFromGroup

Attacher une politique à un groupe IAM

Vous pouvez associer une politique AWS gérée, c'est-à-dire une politique préécrite fournie par, AWSà un groupe d'utilisateurs, comme expliqué dans les étapes suivantes. Pour attacher une politique gérée par le client, c'est-à-dire une politique pour laquelle vous avez créé des autorisations personnalisées, vous devez d'abord créer la politique. Pour plus d'informations sur la création de politiques gérées par le client, consultez Création de politiques IAM.

Pour plus d'informations sur les autorisations et les politiques, consultez Gestion de l'accès aux AWS ressources.

Pour attacher une politique à un groupe d'utilisateurs (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), puis choisissez le nom du groupe.
3. Choisissez l’onglet Permissions (Autorisations).
4. Choisissez Ajouter des autorisations, puis choisissez Joindre des politiques.
5. Les politiques actuellement attachées au groupe d'utilisateurs s'affichent dans la liste Politiques d'autorisation actuelles. Dans la liste des Autres politiques d'autorisation, cochez la case en

regard des noms des politiques à attacher. Vous pouvez utiliser la barre de recherche pour filtrer la liste des politiques par type et nom de politique.

6. Sélectionnez la politique que vous souhaitez associer à votre groupe d'utilisateurs IAM et choisissez Attacher des politiques.

Pour associer une politique à un groupe d'utilisateurs (AWS CLI ou à une AWS API)

Effectuez l'une des actions suivantes :

- AWS CLI: cime attach-group-policy
- AWS API : AttachGroupPolicy

Affectation d'un nouveau nom à un groupe IAM

Lorsque vous modifiez le nom ou le chemin d'accès d'un groupe d'utilisateurs, voici ce qui suit se produit :

- Toutes les politiques attachées au groupe d'utilisateurs restent dans le groupe sous le nouveau nom.
- Le groupe d'utilisateurs conserve également tous ses utilisateurs sous le nouveau nom.
- L'ID unique du groupe d'utilisateurs demeure le même. Pour plus d'informations sur les ID uniques, consultez Identifiants uniques.

IAM ne met pas automatiquement à jour les politiques qui font référence au groupe d'utilisateurs en tant que ressource de manière à utiliser le nouveau nom. Par conséquent, vous devez agir avec prudence lorsque vous renommez un groupe d'utilisateurs. Avant de renommer votre groupe d'utilisateurs, vous devez vérifier manuellement toutes vos politiques afin d'identifier les politiques dans lesquelles le nom de ce groupe d'utilisateurs est mentionné. Par exemple, supposons que Bob est responsable des tests pour l'organisation. Bob dispose d'une politique attachée à son entité d'utilisateur IAM lui permettant d'ajouter des utilisateurs au groupe d'utilisateurs Test et d'en supprimer. Si un administrateur renomme le groupe d'utilisateurs (ou modifie le chemin d'accès au groupe), l'administrateur doit également mettre à jour la politique attachée à Bob afin d'utiliser le nouveau nom ou chemin d'accès. Sinon Bob ne sera plus en mesure d'ajouter ou de supprimer des utilisateurs du groupe d'utilisateurs.

Pour rechercher des politiques qui font référence à un groupe d'utilisateurs en tant que ressource :

1. Dans le panneau de navigation de la console IAM, sélectionnez Policies (Politiques).
2. Triez par la colonne Type pour rechercher vos politiques personnalisées gérées par le client.
3. Sélectionnez le nom de la politique à modifier.
4. Choisissez l'onglet Autorisations, puis Résumé.
5. Choisissez IAM dans la liste des services, si ce service est disponible.
6. Recherchez le nom de votre groupe d'utilisateurs dans la colonne Resource (Ressource).
7. Choisissez Modifier pour changer le nom de votre groupe d'utilisateurs dans la politique.

Pour modifier le nom d'un groupe IAM

Effectuez l'une des actions suivantes :

- AWS Management Console : dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), puis choisissez le nom du groupe. Choisissez Modifier. Tapez le nom du nouveau groupe d'utilisateurs, puis choisissez Save changes (Enregistrer les modifications).
- AWS CLI : aws iam update-group
- AWS API : UpdateGroup

Suppression d'un groupe d'utilisateurs IAM

Lorsque vous supprimez un groupe d'utilisateurs dans le AWS Management Console, la console supprime automatiquement tous les membres du groupe, détache toutes les politiques gérées attachées et supprime toutes les politiques intégrées. Cependant, comme IAM ne supprime pas automatiquement les politiques qui font référence au groupe d'utilisateurs en tant que ressource, vous devez faire attention lorsque vous supprimez un groupe d'utilisateurs. Avant de supprimer votre groupe d'utilisateurs, vous devez vérifier manuellement toutes vos politiques afin d'identifier les politiques dans lesquelles le nom de ce groupe est mentionné. Par exemple, John dispose d'une politique attachée à son entité d'utilisateur IAM lui permettant d'ajouter et supprimer des utilisateurs du groupe d'utilisateurs Test. Si un administrateur supprime le groupe, l'administrateur doit également supprimer la politique attachée à John. Sinon, si l'administrateur recrée le groupe supprimé et lui donne le même nom, les autorisations de John restent en place, même s'il a quitté l'équipe de test.

Pour rechercher des politiques qui font référence à un groupe d'utilisateurs en tant que ressource :

1. Dans le panneau de navigation de la console IAM, sélectionnez Policies (Politiques).
2. Triez par la colonne Type pour rechercher vos politiques personnalisées gérées par le client.
3. Choisissez le nom de politique de la politique à supprimer.
4. Choisissez l'onglet Autorisations, puis Résumé.
5. Choisissez IAM dans la liste des services, si ce service est disponible.
6. Recherchez le nom de votre groupe d'utilisateurs dans la colonne Resource (Ressource).
7. Choisissez Supprimer pour supprimer la politique.
8. Tapez le nom de la politique pour confirmer sa suppression et sélectionnez Supprimer.

En revanche, lorsque vous utilisez les AWS CLI outils pour Windows PowerShell ou AWS l'API pour supprimer un groupe d'utilisateurs, vous devez d'abord supprimer les utilisateurs du groupe. Puis supprimez toutes les politiques en ligne intégrées au groupe d'utilisateurs. Ensuite, détachez toutes les politiques gérées attachées au groupe. Vous pouvez alors supprimer le groupe d'utilisateurs lui-même.

Suppression d'un groupe IAM (Console)

Vous pouvez supprimer un groupe IAM de la AWS Management Console.

Pour supprimer un groupe IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs).
3. Dans la liste des groupes d'utilisateurs, cochez la case en regard du nom des groupes d'utilisateurs à supprimer. Vous pouvez utiliser la zone de recherche pour filtrer la liste des groupes d'utilisateurs par type, autorisations et nom de groupe d'utilisateurs.
4. Sélectionnez Delete (Supprimer).
5. Dans la zone de confirmation, si vous voulez supprimer un seul groupe d'utilisateurs, tapez son nom et choisissez Delete (Supprimer). Si vous voulez supprimer plusieurs groupes d'utilisateurs, tapez le nombre de groupes d'utilisateurs à supprimer suivi de **`user groups`** et choisissez Delete (Supprimer). Par exemple, si vous supprimez trois groupes d'utilisateurs, saisissez **`3 user groups`**.

Suppression d'un groupe d'utilisateurs IAM (AWS CLI)

Vous pouvez supprimer un groupe IAM de la AWS CLI.

Pour supprimer un groupe IAM (AWS CLI)

1. Supprimez tous les utilisateurs du groupe d'utilisateurs.
   - aws iam get-group (pour obtenir la liste des utilisateurs du groupe d'utilisateurs) et aws iam remove-user-from-group (pour supprimer un utilisateur du groupe d'utilisateurs)
2. Supprimez toutes les politiques en ligne intégrées au groupe d'utilisateurs.
   - aws iam list-group-policies (pour obtenir une liste des politiques intégrées du groupe d'utilisateurs) et aws iam delete-group-policy (pour supprimer les politiques en ligne du groupe d'utilisateurs)
3. Détachez toutes les politiques gérées attachées au groupe d'utilisateurs.
   - aws iam list-attached-group-policies (pour obtenir la liste des politiques gérées associées au groupe d'utilisateurs) et aws iam detach-group-policy (pour détacher une politique gérée du groupe d'utilisateurs)
4. Supprimez le groupe d'utilisateurs.
   - aws iam delete-group

Supprimer un groupe d'utilisateurs IAM (AWS API)

Vous pouvez utiliser l' AWS API pour supprimer un groupe d'utilisateurs IAM.

Pour supprimer un groupe d'utilisateurs IAM (AWS API)

1. Supprimez tous les utilisateurs du groupe d'utilisateurs.
   - GetGroup(pour obtenir la liste des utilisateurs du groupe d'utilisateurs) et RemoveUserFromGroup(pour supprimer un utilisateur du groupe d'utilisateurs)
2. Supprimez toutes les politiques en ligne intégrées au groupe d'utilisateurs.
   - ListGroupPolicies(pour obtenir une liste des politiques intégrées du groupe d'utilisateurs) et DeleteGroupPolicy(pour supprimer les politiques intégrées du groupe d'utilisateurs)
3. Détachez toutes les politiques gérées attachées au groupe d'utilisateurs.

- ListAttachedGroupPolicies(pour obtenir une liste des politiques gérées associées au groupe d'utilisateurs) et DetachGroupPolicy(pour détacher une politique gérée du groupe d'utilisateurs)

4. Supprimez le groupe d'utilisateurs.

   - DeleteGroup

Rôles IAM

Un rôle IAM est une identité IAM que vous pouvez créer dans votre compte et qui dispose d'autorisations spécifiques. Un rôle IAM est similaire à un utilisateur IAM, dans la mesure où il s'agit d'une AWS identité dotée de politiques d'autorisation qui déterminent ce que l'identité peut et ne peut pas faire dans ce domaine. AWS En revanche, au lieu d'être associé de manière unique à une personne, un rôle est conçu pour être endossé par tout utilisateur qui en a besoin. En outre, un rôle ne dispose pas d'informations d'identification standard à long terme comme un mot de passe ou des clés d'accès associées. Au lieu de cela, lorsque vous adoptez un rôle, il vous fournit des informations d'identification de sécurité temporaires pour votre session de rôle.

Vous pouvez utiliser des rôles pour déléguer l'accès à des utilisateurs, à des applications ou à des services qui n'ont normalement pas accès à vos AWS ressources. Par exemple, vous pouvez autoriser les utilisateurs de votre AWS compte à accéder à des ressources dont ils ne disposent pas habituellement, ou autoriser les utilisateurs d'un compte à Compte AWS accéder aux ressources d'un autre compte. Vous pouvez également autoriser une application mobile à utiliser AWS des ressources, mais ne pas intégrer de AWS clés dans l'application (où elles peuvent être difficiles à mettre à jour et où les utilisateurs peuvent éventuellement les extraire). Parfois, vous souhaitez donner AWS accès à des utilisateurs dont l'identité est déjà définie à l'extérieur AWS, par exemple dans le répertoire de votre entreprise. Ou, vous pouvez accorder l'accès à votre compte à des tiers afin de leur permettre de réaliser un audit de vos ressources.

Pour ces scénarios, vous pouvez déléguer l'accès aux AWS ressources à l'aide d'un rôle IAM. Cette section présente les rôles et les différentes façons de les utiliser. Elle explique également quand et comment choisir les diverses approches et comment créer, gérer, prendre (endosser) et supprimer des rôles.

> **Note**
>
> Lorsque vous créez votre Compte AWS, aucun rôle n'est créé par défaut. Au fur et à mesure que vous ajoutez des services à votre compte, ils peuvent ajouter des rôles liés à un service pour répondre à leurs cas d'utilisation.
> Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés à un service apparaissent dans votre Compte AWS fichier et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.
> Avant de pouvoir supprimer les rôles liés à un service, vous devez d'abord supprimer les ressources qui leur sont associées. Vos ressources sont ainsi protégées, car vous ne pouvez pas involontairement supprimer l'autorisation d'accéder aux ressources.
> Pour plus d'informations concernant la prise en charge par les services des rôles liés à un service, consultez AWS services qui fonctionnent avec IAM et recherchez les services affichant Oui dans la colonne Rôle lié à un service. Choisissez un Yes (Oui) ayant un lien permettant de consulter les détails du rôle pour ce service.

Rubriques

- Termes et concepts relatifs aux rôles
- Scénarios courants pour les rôles : utilisateurs, applications et services
- Utilisation des rôles liés à un service
- Création de rôles IAM
- Utilisation de rôles IAM
- Gestion des rôles IAM

Termes et concepts relatifs aux rôles

Voici quelques termes de base pour vous aider à vous familiariser avec les rôles.

Rôle

Une identité IAM que vous pouvez créer dans votre compte et qui dispose d'autorisations spécifiques. Un rôle IAM présente des similitudes avec un utilisateur IAM. Les rôles et les utilisateurs sont tous deux des identités AWS avec des politiques d'autorisations qui déterminent

ce que l'identité peut ou ne peut pas faire dans AWS. En revanche, au lieu d'être associé de manière unique à une personne, un rôle est conçu pour être endossé par tout utilisateur qui en a besoin. En outre, un rôle ne dispose pas d'informations d'identification standard à long terme comme un mot de passe ou des clés d'accès associées. Au lieu de cela, lorsque vous adoptez un rôle, il vous fournit des informations d'identification de sécurité temporaires pour votre session de rôle.

Les rôles peuvent être utilisés par :

- Un utilisateur IAM ayant le même Compte AWS rôle
- Un utilisateur IAM dont le rôle Compte AWS est différent
- Un service Web proposé par AWS Amazon Elastic Compute Cloud (Amazon EC2)
- Un utilisateur externe authentifié par un service de fournisseur d'identité (IdP) externe, compatible avec SAML 2.0 ou OpenID Connect, ou un broker d'identité personnalisé.

AWS rôle de service

Une fonction de service est un rôle IAM qu'un service endosse pour accomplir des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer une fonction du service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.

AWS rôle de service pour une instance EC2

Un genre particulier de rôle de service qu'une application s'exécutant sur une instance Amazon EC2 peut endosser pour effectuer des actions dans votre compte. Ce rôle est attribué à l'instance EC2 lorsque vous la lancez. Des applications exécutées sur cette instance peuvent récupérer des informations d'identification de sécurité temporaires et effectuer des actions permises par le rôle. Pour plus d'informations sur l'utilisation d'un rôle de service pour une instance EC2, consultez Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2.

AWS rôle lié au service

Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés au service apparaissent dans votre Compte AWS fichier et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

> **Note**
>
> Si vous utilisez déjà un service lorsqu'il commence à prendre en charge des rôles liés à un service, vous pouvez recevoir un e-mail vous informant de l'existence d'un nouveau rôle sur votre compte. Dans ce cas, le service crée automatiquement le rôle lié à un service sur votre compte. Aucune action de votre part n'est requise pour prendre ce rôle en charge et il est préférable de ne pas le supprimer manuellement. Pour plus d'informations, veuillez consulter Un nouveau rôle est apparu dans mon compte AWS.

Pour plus d'informations concernant la prise en charge par les services des rôles liés à un service, consultez AWS services qui fonctionnent avec IAM et recherchez les services affichant Oui dans la colonne Rôle lié à un service. Choisissez un Yes (Oui) ayant un lien permettant de consulter les détails du rôle pour ce service. Pour plus d'informations, consultez Utilisation des rôles liés à un service.

Création de chaînes de rôles

Le chaînage de rôles consiste à utiliser un rôle pour assumer un second rôle via l'API AWS CLI or. Par exemple, `RoleA` dispose de l'autorisation d'assumer `RoleB`. Vous pouvez autoriser User1 à assumer `RoleA` en utilisant ses informations d'identification utilisateur à long terme dans le cadre du fonctionnement de l' AssumeRole API. Cette opération renvoie les informations d'identification à court terme de `RoleA`. Avec le chaînage de rôles, vous pouvez utiliser les informations d'identification à court terme de `RoleA` pour permettre à User1 d'assumer `RoleB`.

Lorsque vous endossez un rôle, vous pouvez passer une balise de session et la définir comme transitive. Les balises de session transitives sont transmises à toutes les sessions suivantes d'une chaîne de rôles. Pour en savoir plus sur les balises de session, veuillez consulter Transmission des balises de session AWS STS.

Le chaînage des rôles limite votre session de rôle AWS CLI ou celle de l' AWS API à une heure maximum. Lorsque vous utilisez l'opération AssumeRoled'API pour assumer un rôle, vous pouvez spécifier la durée de votre session de rôle à l'aide du `DurationSeconds` paramètre. Vous pouvez spécifier une valeur de paramètre jusqu'à 43200 secondes (12 heures), en fonction du paramètre de durée de session maximale de votre rôle. Toutefois, si vous endossez un rôle à l'aide de la création de chaînes de rôles et que vous définissez une valeur de paramètre `DurationSeconds` supérieure à une heure, l'opération échoue.

AWS ne considère pas l'utilisation de rôles pour accorder des autorisations à des applications qui s'exécutent sur des instances EC2 comme un chaînage de rôles.

Délégation

L'octroi à une personne d'autorisations lui permettant d'accéder aux ressources que vous contrôlez. La délégation implique la mise en place d'une confiance entre deux comptes. Le premier est le compte propriétaire de la ressource (le compte de confiance). Le second est le compte qui contient les utilisateurs qui doivent accéder à la ressource (le compte approuvé). Les comptes d'approbation et approuvés peuvent être :

- Un même compte.
- Comptes distincts se trouvant tous deux sous le contrôle de votre organisation.
- Deux comptes appartenant à des organisations différentes.

Pour déléguer l'autorisation d'accès à une ressource, vous créez un rôle IAM dans le compte d'approbation auquel sont attachées deux politiques. La politique d'autorisation accorde à l'utilisateur du rôle les autorisations nécessaires pour exécuter les tâches prévues sur la ressource. La politique d'approbation détermine les membres du compte autorisés à endosser le rôle.

Lorsque vous créez une politique d'approbation, vous ne pouvez pas spécifier de caractère générique (*) comme ARN dans l'élément de principal. La politique d'approbation est attachée au rôle dans le compte d'approbation, et représente la moitié des autorisations. L'autre moitié est une politique d'autorisation attachée à l'utilisateur dans le compte approuvé qui autorise cet utilisateur à prendre ou endosser le rôle. Un utilisateur qui endosse un rôle temporairement abandonne ses propres autorisations, de manière à adopter celles du rôle. Lorsque l'utilisateur quitte le rôle ou cesse de l'utiliser, ses autorisations d'origine sont restaurées. Un paramètre supplémentaire appelé ID externe permet de sécuriser l'utilisation de rôles entre des comptes qui ne sont pas contrôlés par la même organisation.

Fédération

Création d'une relation de confiance entre un fournisseur d'identité externe et AWS. Les utilisateurs peuvent se connecter à un fournisseur OIDC, tel que Login with Amazon, Facebook, Google ou tout autre IdP compatible avec OpenID Connect (OIDC). Ils peuvent également se connecter à un système d'identité d'entreprise compatible avec SAML (Security Assertion Markup Language) 2.0 tel que les services ADFS (Active Directory Federation Services) de Microsoft. Lorsque vous utilisez OIDC et SAML 2.0 pour configurer une relation de confiance entre ces fournisseurs d'identité externes AWS, un rôle IAM est attribué à l'utilisateur. L'utilisateur reçoit

également des informations d'identification temporaires qui lui permettent d'accéder à vos AWS ressources.

Utilisateur fédéré

Au lieu de créer un utilisateur IAM, vous pouvez utiliser des identités existantes provenant du AWS Directory Service répertoire des utilisateurs de votre entreprise ou d'un fournisseur OIDC. Ils sont appelés utilisateurs fédérés. AWS attribue un rôle à un utilisateur fédéré lorsque l'accès est demandé par le biais d'un fournisseur d'identité. Pour de plus amples informations sur les utilisateurs fédérés, veuillez consulter Utilisateurs fédérés et rôles.

Politique d'approbation

Document de politique JSON dans lequel vous définissez les principaux en lesquels vous avez confiance pour endosser le rôle. Une politique d'approbation de rôle est une politique basée sur les ressources requise qui est attachée à un rôle dans IAM. Les principaux que vous pouvez spécifier dans la politique d'approbation comprennent les utilisateurs, les rôles, les comptes et les services.

Politique d'autorisations

Un document d'autorisations au format JSON dans lequel vous définissez les actions et ressources que le rôle peut utiliser. Le document est écrit conformément aux règles du langage de politique IAM.

Limite d'autorisations

Une fonction avancée dans laquelle vous utilisez des politiques pour limiter les autorisations maximales qu'une politique basée sur les identités peut accorder à un rôle. Vous ne pouvez pas appliquer une limite d'autorisations à un rôle lié à un service. Pour plus d’informations, veuillez consulter Limites d'autorisations pour les entités IAM.

Principal

Entité AWS capable d'effectuer des actions et d'accéder aux ressources. Un principal peut être un Utilisateur racine d'un compte AWS utilisateur IAM ou un rôle. Vous pouvez accorder des autorisations d'accès à une ressource de l'une des façons suivantes :

- Vous pouvez attacher une politique d'autorisation à un utilisateur (directement, ou indirectement via un groupe) ou à un rôle.
- Pour les services qui prennent en charge les politiques basées sur les ressources, vous pouvez identifier le principal dans l'élément `Principal` d'une politique attachée à la ressource.

Si vous faites référence à un Compte AWS comme principal, cela signifie généralement tout principal défini dans ce compte.

> **Note**
>
> Vous ne pouvez pas utiliser un caractère générique (*) pour faire correspondre une partie d'un nom de principal ou d'un ARN dans une politique d'approbation d'un rôle. Pour plus de détails, consultez AWS Éléments de politique JSON : Principal.

Rôle pour l'accès entre comptes

Un rôle qui octroie l'accès aux ressources d'un compte à un principal approuvé d'un autre compte. Les rôles constituent le principal moyen d'accorder l'accès intercompte. Toutefois, certains services AWS vous permettent d'attacher une politique directement à une ressource (au lieu d'utiliser un rôle en tant que proxy). Ces politiques sont appelées politiques basées sur les ressources, et vous pouvez les utiliser pour accorder aux principaux un autre Compte AWS accès à la ressource. Ces ressources incluent notamment les compartiments Amazon Simple Storage Service (S3), les coffres S3 Glacier, les rubriques Amazon Simple Notification Service (SNS) et les files d'attente Amazon Simple Queue Service (SQS). Pour connaître les services qui prennent en charge les politiques basées sur les ressources, veuillez consulter AWS services qui fonctionnent avec IAM. Pour plus d'informations sur les politiques basées sur les ressources, consultez Accès intercompte aux ressources dans IAM.

Scénarios courants pour les rôles : utilisateurs, applications et services

Comme pour la plupart des AWS fonctionnalités, vous pouvez généralement utiliser un rôle de deux manières : de manière interactive dans la console IAM ou par programmation avec les outils pour Windows PowerShell ou l' AWS CLI API.

- Les utilisateurs IAM de votre compte qui utilisent la console IAM peuvent basculer vers un rôle leur permettant d'utiliser temporairement les autorisation du rôle dans la console. Les utilisateurs abandonnent leurs autorisations d'origine et acceptent les autorisations attribuées au rôle. Lorsque les utilisateurs quittent le rôle, leurs autorisations d'origine sont restaurées.
- Une application ou un service proposé par AWS (comme Amazon EC2) peut assumer un rôle en demandant des informations d'identification de sécurité temporaires pour un rôle auquel envoyer des demandes programmatiques. AWS Vous utilisez ce rôle ainsi pour éviter de partager ou de

conserver des informations d'identification de sécurité à long terme (par exemple, en créant un utilisateur IAM) pour chaque entité qui doit accéder à une ressource.

> **Note**
>
> Ce manuel utilise les phrases passer à un rôle et endosser un rôle de manière interchangeable.

La forme d'utilisation des rôles la plus simple consiste à accorder à vos utilisateurs IAM l'autorisation de basculer vers des rôles que vous créez au sein de votre propre entreprise ou d'une autre Compte AWS. Ils peuvent changer de rôles facilement à l'aide de la console IAM pour utiliser des autorisations dont vous ne souhaitez pas qu'ils disposent d'ordinaire et il leur suffit de quitter le rôle pour renoncer à ces autorisations. Cela permet d'empêcher un accès accidentel à des ressources sensibles ou leur modification involontaire.

Pour utiliser les rôles de manière plus complexe, comme accorder l'accès à des applications et des services, ou à des utilisateurs externes fédérés, vous pouvez appeler l'API `AssumeRole`. Cet appel d'API renvoie un ensemble d'informations d'identification temporaires que l'application peut utiliser dans les appels d'API suivants. Les tentatives d'actions avec les informations d'identification temporaires ne disposent que des autorisations accordées par le rôle associé. Une application n'a pas besoin de « quitter » le rôle de la même manière qu'un utilisateur dans la console. L'application arrête simplement d'utiliser les informations d'identification temporaires et reprend ses appels avec les informations d'identification d'origine.

Les utilisateurs fédérés se connectent à l'aide des informations d'identification d'un fournisseur d'identité (IdP). AWS fournit ensuite des informations d'identification temporaires à l'IdP de confiance à transmettre à l'utilisateur pour les inclure dans les demandes de AWS ressources ultérieures. Ces informations d'identification fournissent des autorisations accordées au rôle attribué.

Cette section présente les scénarios suivants :

- Donnez accès à un utilisateur IAM dans un compte Compte AWS que vous possédez pour accéder aux ressources d'un autre compte que vous possédez
- Fournir un accès aux charges de travail qui n'appartiennent pas à AWS
- Octroyer un accès aux utilisateurs IAM dans des Comptes AWS appartenant à des tierces parties
- Fournir un accès aux services offerts par AWS deux AWS ressources

- Octroi de l'accès à des utilisateurs authentifiés en externe (fédération d'identité)

Fournir l'accès à un utilisateur IAM dans un autre utilisateur Compte AWS dont vous êtes le propriétaire

Vous pouvez autoriser vos utilisateurs IAM à passer à des rôles au sein de votre entreprise Compte AWS ou à des rôles définis dans d'autres rôles Comptes AWS que vous possédez.

> **Note**
>
> Si vous souhaitez accorder l'accès à un compte que ne vous appartenant pas ou que vous ne contrôlez pas, consultez Fournir un accès à des Comptes AWS sites appartenant à des tiers ultérieurement dans cette rubrique.

Imaginons que vous disposez d'instances Amazon EC2 qui sont critiques pour votre organisation. Au lieu d'accorder directement à vos utilisateurs l'autorisation de supprimer les instances, vous pouvez créer un rôle avec ces privilèges. Ensuite, autorisez les administrateurs à passer au rôle lorsqu'ils ont besoin de supprimer une instance. Cela ajoute les couches de protection suivantes aux instances :

- Vous devez accorder explicitement à vos utilisateurs l'autorisation d'endosser le rôle.
- Vos utilisateurs doivent passer activement au rôle à l'aide de l'API AWS Management Console or ou assumer le rôle à l'aide de l' AWS API AWS CLI or.
- Vous pouvez ajouter une protection d'authentification multi-facteur (MFA) au rôle afin que seuls les utilisateurs qui se connectent avec un dispositif MFA puissent endosser le rôle. Pour apprendre à configurer un rôle afin que les utilisateurs qui l'endossent doivent d'abord être authentifiés à l'aide de l'authentification multi-facteur (MFA), consultez Configuration de l'accès aux API protégé par MFA.

Nous vous recommandons d'utiliser cette approche pour appliquer le principe du moindre privilège. Cela implique de restreindre l'utilisation des autorisations d'un niveau élevé aux seules fois où elles sont requises pour des tâches spécifiques. Grâce aux rôles, vous pouvez empêcher les modifications accidentelles apportées aux environnements sensibles, en particulier si vous les combinez à des audits pour vous assurer que les rôles sont utilisés uniquement quand ils sont requis.

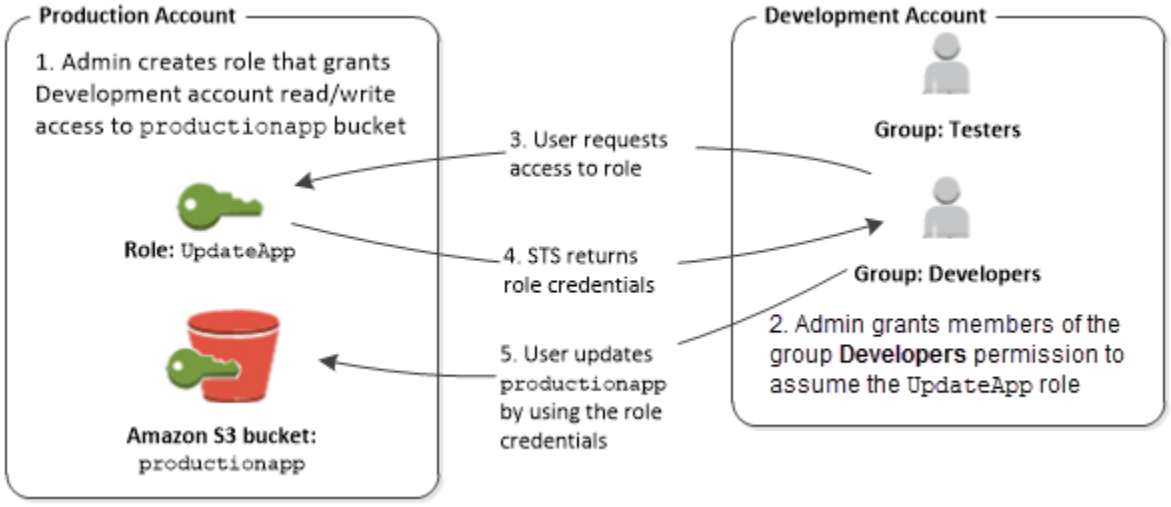
Lorsque vous créez un rôle à cette fin, vous spécifiez les comptes en fonction de l'ID des utilisateurs ayant besoin d'accéder à l'élément `Principal` de la politique d'approbation du rôle. Vous pouvez

ensuite accorder à des utilisateurs spécifiques de ces autres comptes des autorisations à passer au rôle. Pour savoir si les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, consultez Qu'est-ce qu'IAM Access Analyzer ?.

Un utilisateur d'un compte peut passer à un rôle dans le même compte ou dans un autre compte. Lorsqu'il utilise le rôle, l'utilisateur peut exécuter uniquement les actions et accéder uniquement aux ressources autorisées par le rôle. Leurs autorisations utilisateur d'origine sont suspendues. Lorsque l'utilisateur quitte le rôle, ses autorisations utilisateur d'origine sont restaurées.

Exemple de scénario utilisant des comptes de développement et de production distincts

Imaginez que votre entreprise en dispose de plusieurs Comptes AWS pour isoler un environnement de développement d'un environnement de production. Les utilisateurs du compte de développement peuvent parfois avoir besoin d'accéder aux ressources du compte de production. Par exemple, vous pouvez avoir besoin d'un accès entre comptes lorsque vous promouvez une mise à jour depuis l'environnement de développement vers l'environnement de production. Même si vous pouvez créer des identités (et mots de passe) distinctes pour les utilisateurs qui travaillent dans les deux comptes, la gestion des informations d'identification pour plusieurs comptes rend la gestion des identités difficile. Dans l'illustration suivante, tous les utilisateurs sont gérés dans le compte de développement, mais certains développeurs nécessitent un accès limité au compte de production. Le compte de développement se compose de deux groupes : les Développeurs et les Testeurs, chacun disposant de sa propre politique.



1. Dans le compte de production, un administrateur utilise IAM pour créer le rôle `UpdateApp` dans ce compte. Dans le rôle, l'administrateur définit une politique d'approbation qui spécifie le compte

de développement en tant que `Principal`, ce qui signifie que les utilisateurs autorisés du compte de développement peuvent utiliser le rôle `UpdateApp`. L'administrateur définit également une politique d'autorisations pour le rôle qui spécifie les autorisations de lecture et d'écriture sur le compartiment Amazon S3 appelé `productionapp`.

L'administrateur partage ensuite les informations appropriées avec toute personne qui doit endosser le rôle. Ces informations sont le numéro de compte et le nom du rôle (pour les utilisateurs de AWS console) ou le nom de ressource Amazon (ARN) (pour AWS CLI AWS l'accès à l'API). L'ARN du rôle peut ressembler à `arn:aws:iam::123456789012:role/UpdateApp`, où le rôle est appelé `UpdateApp`. Le rôle a été créé dans le compte dont le numéro est 123456789012.

> **Note**
>
> L'administrateur peut, facultativement, configurer le rôle afin que les utilisateurs qui endossent le rôle doivent d'abord être authentifiés à l'aide de l'authentification multi-facteur (MFA). Pour plus d'informations, veuillez consulter Configuration de l'accès aux API protégé par MFA.

2. Dans le compte de développement, un administrateur accorde aux membres du groupe Développeurs l'autorisation de changer de rôle. Cela se fait en accordant au groupe de développeurs l'autorisation d'appeler l'`AssumeRole`API AWS Security Token Service (AWS STS) pour le `UpdateApp` rôle. Tout utilisateur IAM appartenant au groupe Développeurs dans le compte de développement peut à présent basculer vers le rôle `UpdateApp` du compte de production. Les autres utilisateurs n'appartenant pas au groupe de développeurs n'ont pas l'autorisation de passer au rôle et ne peuvent, donc, pas accéder au compartiment S3 dans le compte de production.
3. L'utilisateur demande les changements de rôle :
   - AWS console : l'utilisateur choisit le nom du compte dans la barre de navigation et choisit Switch Role. L'utilisateur spécifie l'ID (ou l'alias) du compte, ainsi que le nom du rôle. Sinon, l'utilisateur peut cliquer sur un lien envoyé par e-mail par l'administrateur. Le lien dirige l'utilisateur vers la page Changer de rôle avec les détails déjà remplis.
   - AWS API/AWS CLI : Un utilisateur du groupe Développeurs du compte de développement appelle la `AssumeRole` fonction pour obtenir les informations d'identification du `UpdateApp` rôle. L'utilisateur spécifie l'ARN du rôle `UpdateApp` dans le cadre de l'appel. Si un utilisateur du groupe Testeurs fait la même demande, la demande échoue, car les Testeurs ne sont pas autorisés à appeler `AssumeRole` pour l'ARN de rôle `UpdateApp`.

4. AWS STS renvoie des informations d'identification temporaires :
   - AWS console : AWS STS vérifie la demande avec la politique de confiance du rôle pour s'assurer que la demande provient d'une entité de confiance (il s'agit du compte de développement). Après vérification, AWS STS renvoie les informations d'identification de sécurité temporaires à la AWS console.
   - API/CLI : AWS STS vérifie la demande par rapport à la politique de confiance du rôle pour s'assurer que la demande provient d'une entité de confiance (il s'agit du compte de développement). Après vérification, AWS STS renvoie les informations de sécurité temporaires à l'application.
5. Les informations d'identification temporaires permettent d'accéder à la AWS ressource :
   - AWS console : la AWS console utilise les informations d'identification temporaires au nom de l'utilisateur pour toutes les actions de console suivantes, dans ce cas, pour lire et écrire dans le `productionapp` compartiment. La console ne peut pas accéder à d'autres ressources du compte de production. Lorsque l'utilisateur quitte le rôle, les autorisations dont ils disposait avant de changer de rôle sont restaurées.
   - API/CLI : L'application utilise les informations d'identification de sécurité temporaires pour mettre à jour le compartiment `productionapp`. Avec les informations d'identification de sécurité temporaires, l'application peut uniquement lire et écrire dans le compartiment `productionapp`, mais ne peut pas accéder à d'autres ressources du compte Production. L'application n'a pas besoin de quitter le rôle. Au contraire, elle arrête d'utiliser les informations d'identification temporaires et utilise les informations d'identification d'origine dans les appels d'API suivants.

En savoir plus

Pour plus d'informations, consultez les ressources suivantes :

- Didacticiel IAM : déléguer l'accès entre des comptes AWS à l'aide des rôles IAM

Fournir un accès aux non AWS charges de travail

Un rôle IAM est un objet dans AWS Identity and Access Management (IAM) auquel des autorisations sont attribuées. Lorsque vous assumez ce rôle en utilisant une identité IAM ou une identité extérieure AWS, cela vous fournit des informations d'identification de sécurité temporaires pour votre session de rôle. Il se peut que des charges de travail s'exécutent dans votre centre de données ou dans une autre infrastructure en dehors de AWS ce besoin d'accéder à vos AWS ressources. Au lieu de créer, de distribuer et de gérer des clés d'accès à long terme, vous pouvez utiliser AWS Identity and Access

Management Roles Anywhere (IAM Roles Anywhere) pour authentifier vos applications non AWS liées à des charges de travail. IAM Roles Anywhere utilise les certificats X.509 de votre autorité de certification (CA) pour authentifier les identités et fournir un accès sécurisé à l' Services AWS aide des informations d'identification temporaires fournies par un rôle IAM.

Pour utiliser IAM Roles Anywhere, vous devez configurer une autorité de certification à l'aide de AWS Private Certificate Authority ou utiliser une autorité de certification issue de votre propre infrastructure PKI. Après avoir configuré une autorité de certification, vous créez un objet dans IAM Roles Anywhere appelé Ancre de confiance pour établir une relation de confiance entre IAM Roles Anywhere et votre autorité de certification pour l'authentification. Vous pouvez ensuite configurer vos rôles IAM existants ou créer de nouveaux rôles qui font confiance au service IAM Roles Anywhere. Lorsque vos personnes autres que les AWS charges de travail s'authentifient auprès d'IAM Roles Anywhere à l'aide de l'ancre de confiance, elles peuvent obtenir des informations d'identification temporaires pour que vos rôles IAM puissent accéder à vos ressources. AWS

Pour plus d'informations sur IAM Roles Anywhere, consultez Présentation de AWS Identity and Access Management Rôle Anywhere dans le Guide de l'utilisateur d'IAM Roles Anywhere.

Fournir un accès à des Comptes AWS sites appartenant à des tiers

Lorsque des tiers ont besoin d'accéder aux AWS ressources de votre organisation, vous pouvez utiliser des rôles pour leur déléguer l'accès. Par exemple, un tiers peut fournir un service de gestion de vos ressources AWS . Avec les rôles IAM, vous pouvez accorder à ces tiers l'accès à vos AWS ressources sans partager vos informations d'identification AWS de sécurité. Au lieu de cela, le tiers peut accéder à vos AWS ressources en assumant un rôle que vous créez dans votre Compte AWS. Pour savoir si les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, consultez Qu'est-ce qu'IAM Access Analyzer ?.

Les tiers doivent vous fournir les informations suivantes pour vous permettre de créer un rôle qu'ils peuvent endosser :

- L' Compte AWS identifiant du tiers. Vous spécifiez leur Compte AWS identifiant en tant que principal lorsque vous définissez la politique de confiance pour le rôle.
- Un ID externe à attacher uniquement à ce rôle. L'ID externe peut être n'importe quel identifiant qui n'est connu que de vous et de la tierce partie. Par exemple, vous pouvez utiliser un ID de facture entre les tiers et vous-même, mais n'utilisez aucun élément pouvant être deviné, tels que le nom ou le numéro de téléphone du tiers. Vous devez spécifier cet ID lorsque vous définissez la politique d'approbation pour le rôle. Les tiers doivent fournir cette ID lorsqu'ils endosser le rôle. Pour plus

d'informations sur l'ID externe, consultez Comment utiliser un identifiant externe lorsque vous accordez l'accès à vos AWS ressources à un tiers.

- Les autorisations dont le tiers a besoin pour utiliser vos AWS ressources. Vous devez spécifier ces autorisations lors de la définition de la politique d'autorisation du rôle. Cette politique définit les actions que les utilisateurs tiers peuvent entreprendre et les ressources auxquelles ils peuvent accéder.

Après avoir créé le rôle, vous devez fournir l'Amazon Resource Name (ARN) du rôle au tiers. Il a besoin de l'ARN de votre rôle pour endosser le rôle.

> **Important**
>
> Lorsque vous accordez à des tiers l'accès à vos AWS ressources, ils peuvent accéder à toutes les ressources que vous spécifiez dans la politique. L'utilisation de vos ressources par ces tiers vous est facturée. Veillez à limiter leur utilisation de vos ressources de façon appropriée.

Comment utiliser un identifiant externe lorsque vous accordez l'accès à vos AWS ressources à un tiers

Vous devez parfois autoriser un tiers à accéder à vos AWS ressources (accès délégué). Un aspect important de ce scénario est l'ID externe, informations facultatives que vous pouvez utiliser dans une politique d'approbation des rôles IAM afin de désigner l'utilisateur autorisé à endosser le rôle.

> **Important**
>
> AWS ne traite pas l'identifiant externe comme un secret. Une fois que vous avez créé un secret, tel qu'une paire de clés d'accès ou un mot de passe AWS, vous ne pouvez plus le consulter. L'ID externe d'un rôle peut être vu par n'importe qui ayant l'autorisation de consulter le rôle.

Dans un environnement mutualisé où vous prenez en charge plusieurs clients avec différents AWS comptes, nous vous recommandons d'utiliser un identifiant externe par Compte AWS client. Cet ID doit être une chaîne aléatoire générée par le tiers.

Pour exiger que le tiers fournisse un ID externe lors de la prise en charge d'un rôle, mettez à jour la politique d'approbation du rôle avec l'ID externe de votre choix.

Pour fournir un identifiant externe lorsque vous assumez un rôle, utilisez l' AWS API AWS CLI or pour assumer ce rôle. Pour plus d'informations, consultez l'opération de l'AssumeRoleAPI STS ou l'opération de la CLI STS assume-role.

Supposons, par exemple, que vous décidiez de faire appel à une société tierce appelée Example Corp pour surveiller vos coûts Compte AWS et vous aider à optimiser les coûts. Afin de suivre vos dépenses quotidiennes, Example Corp doit accéder à vos AWS ressources. Example Corp surveille également de nombreux autres comptes AWS pour d'autres clients.

Ne donnez pas à Exemple Corp l'accès à un utilisateur IAM et à ses informations d'identification à long terme dans votre compte AWS . Utilisez plutôt un rôle IAM et ses informations d'identification de sécurité temporaires. Un rôle IAM fournit un mécanisme permettant à un tiers d'accéder à vos AWS ressources sans avoir à partager des informations d'identification à long terme (telles qu'une clé d'accès utilisateur IAM).

Vous pouvez utiliser un rôle IAM pour établir une relation de confiance entre votre compte Compte AWS et le compte Example Corp. Une fois cette relation établie, un membre du compte Example Corp peut appeler l' AWS Security Token Service AssumeRoleAPI pour obtenir des informations d'identification de sécurité temporaires. Les membres d'Example Corp peuvent ensuite utiliser les informations d'identification pour accéder aux AWS ressources de votre compte.

> **Note**
>
> Pour plus d'informations sur les opérations d' AWS API AssumeRole et sur les autres opérations que vous pouvez appeler pour obtenir des informations d'identification de sécurité temporaires, consultezDemande d'informations d'identification temporaires de sécurité.

Voici comment s'articule ce scénario :

1. Vous embauchez Example Corp qui crée un identifiant client unique pour vous. Ils vous fournissent cet identifiant client unique et leur Compte AWS numéro. Vous avez besoin de ces informations pour créer un rôle IAM à l'étape suivante.

> **Note**
>
> Example Corp peut utiliser n'importe quelle valeur de chaîne pour le ExternalId, à condition qu'elle soit unique pour chaque client. Il peut s'agir d'un numéro de compte client ou encore d'une chaîne aléatoire de caractères, à condition que chaque client ait une valeur différente. Elle n'est pas censée être « secrète ». Example Corp doit fournir la ExternalId valeur à chaque client. Ce qui compte, c'est qu'elle soit générée par Example Corp et non par ses clients afin de garantir que chaque ID externe est unique.

2. Vous vous connectez AWS et créez un rôle IAM qui permet à Example Corp d'accéder à vos ressources. Comme tout autre rôle IAM, celui-ci dispose de deux politiques, une politique d'autorisation et une politique d'approbation. La politique d'approbation du rôle spécifie qui peut endosser le rôle. Dans notre exemple de scénario, la politique spécifie le Compte AWS nombre d'Example Corp comme étant le`Principal`. Cela permet aux identités de ce compte d'endosser le rôle. En outre, vous ajoutez un élément `Condition` à la politique d'approbation. Cette `Condition` teste la clé de contexte `ExternalId` afin de s'assurer qu'elle correspond à l'ID client unique issu d'Exemple Corp. Exemple :

```
   "Principal": {"AWS": "Example Corp's Compte AWS ID"},
   "Condition": {"StringEquals": {"sts:ExternalId": "Unique ID Assigned by Example
Corp"}}
```

3. La politique d'autorisation du rôle spécifie ce que le rôle permet à un utilisateur de faire. Par exemple, vous pouvez spécifier que le rôle permet à un utilisateur de gérer uniquement vos ressources Amazon EC2 et Amazon RDS, mais pas vos utilisateurs ou groupes IAM. Dans notre exemple de scénario, vous utilisez la politique d'autorisation pour accorder à Example Corp un accès en lecture seule à toutes les ressources de votre compte.
4. Après avoir créé le rôle, vous devez fournir l'Amazon Resource Name (ARN) du rôle à Example Corp.
5. Lorsque Example Corp a besoin d'accéder à vos AWS ressources, un membre de l'entreprise appelle l' AWS `sts:AssumeRole`API. L'appel inclut l'ARN du rôle à assumer et le ExternalId paramètre correspondant à leur identifiant client.

Si la demande provient d'une personne utilisant Example Corp Compte AWS, et si l'ARN du rôle et l'ID externe sont corrects, la demande aboutit. Il fournit ensuite des informations de sécurité

temporaires qu'Example Corp peut utiliser pour accéder aux AWS ressources autorisées par votre rôle.

Autrement dit, quand une politique de rôle inclut un ID externe, tous ceux qui souhaitent endosser le rôle doivent non seulement être spécifiés comme principaux dans le rôle, mais également inclure l'ID externe correct.

Pourquoi utiliser un ID externe ?

De manière abstraite, l'ID externe autorise l'utilisateur qui endosse le rôle d'indiquer les circonstances dans lesquels il travaille. Il permet également au titulaire du compte d'autoriser que le rôle soit endossé uniquement dans des circonstances spécifiques. La fonction principale de l'ID externe consiste à traiter et à prévenir Le problème de l'adjoint confus.

Quand est-il conseillé d'utiliser un ID externe ?

Utilisez un ID externe dans les situations suivantes :

- Vous êtes Compte AWS propriétaire et vous avez configuré un rôle pour un tiers qui accède Comptes AWS à d'autres rôles que le vôtre. Vous devez demander à ce tiers un ID externe qu'il inclura lorsqu'il endossera votre rôle. Ensuite, vous vérifiez cet ID externe dans la politique d'approbation de votre rôle. Ainsi, la tierce partie peut endosser votre rôle uniquement lorsqu'elle agit en votre nom.
- Vous êtes en position d'endosser des rôles pour le compte de différents clients comme Exemple Corp dans notre précédent scénario. Vous devez attribuer un ID externe unique à chaque client et leur demander de l'ajouter à leur politique d'approbation du rôle. Vous devez ensuite vous assurer de toujours inclure l'ID externe correct dans vos demandes pour endosser des rôles.

  Vous disposez probablement déjà d'un identifiant unique pour chacun de vos clients, et cet ID unique est suffisant pour être utilisé comme ID externe. L'ID externe n'est pas une valeur spéciale que vous devez créer de manière explicite, ou suivre séparément, juste à cette fin.

  Vous devez toujours spécifier l'ID externe dans vos appels d'API `AssumeRole`. De plus, lorsqu'un client vous fournit un ARN de rôle, vérifiez que vous pouvez endosser le rôle avec et sans l'ID externe correct. Si vous pouvez endosser le rôle sans l'ID externe approprié, ne stockez pas l'ARN du rôle du client dans votre système. Attendez que le client mette à jour la politique d'approbation du rôle pour demander l'ID externe. Ainsi, vous aidez vos clients à faire ce qu'il faut, ce qui vous permet de vous protéger tous les deux contre le problème du député confus.

Octroi d'accès à un service AWS

De nombreux AWS services nécessitent que vous utilisiez des rôles pour contrôler les accès auxquels ils peuvent accéder. Un rôle qu'un service endosse pour effectuer des actions en votre nom s'appelle un rôle de service. Lorsqu'un rôle de service remplit un objectif spécial pour un service, il peut être défini comme un rôle de service pour les instances EC2 ou un rôle lié à un service. Consultez la documentation AWS spécifique à chaque service pour vérifier s'il utilise des rôles et apprendre à attribuer un rôle au service afin qu'il l'utilise.

Pour plus de détails sur la création d'un rôle permettant de déléguer l'accès à un service proposé par AWS, consultezCréation d'un rôle pour la délégation d'autorisations à un service AWS.

Le problème de l'adjoint confus

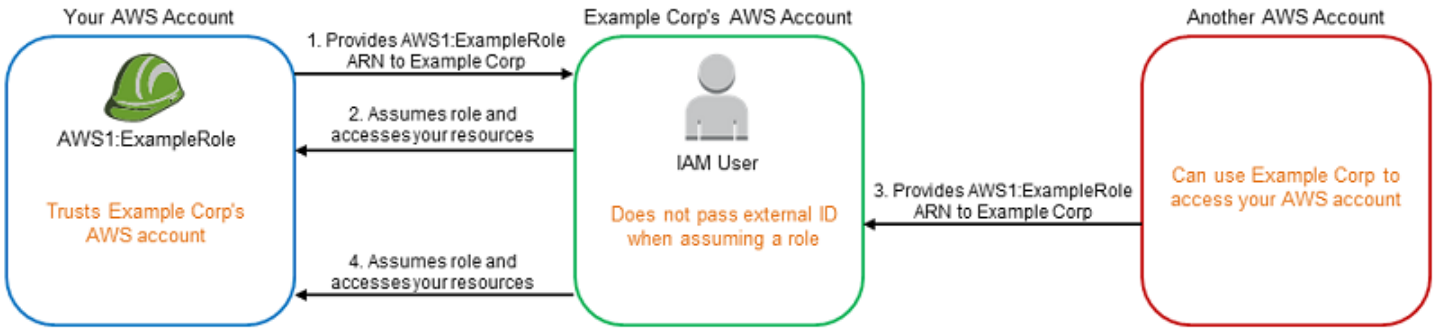
Le problème de député confus est un problème de sécurité dans lequel une entité qui n’est pas autorisée à effectuer une action peut contraindre une entité plus privilégiée à le faire. Pour éviter cela, AWS fournit des outils qui vous aident à protéger votre compte si vous fournissez à des tiers (comptes croisés) ou à d'autres AWS services (appelés interservices) un accès aux ressources de votre compte.

Il peut arriver que vous deviez accorder à un tiers l'accès à vos AWS ressources (accès délégué). Supposons, par exemple, que vous décidiez de faire appel à une société tierce appelée Example Corp pour surveiller vos coûts Compte AWS et vous aider à optimiser les coûts. Afin de suivre vos dépenses quotidiennes, Example Corp doit accéder à vos AWS ressources. Example Corp en surveille également de nombreux autres Comptes AWS pour le compte d'autres clients. Vous pouvez utiliser un rôle IAM pour établir une relation de confiance entre votre compte Compte AWS et le compte Example Corp. Un aspect important de ce scénario est l'ID externe, informations facultatives que vous pouvez utiliser dans une politique d'approbation des rôles IAM afin de désigner l'utilisateur autorisé à endosser le rôle. La fonction principale de l'ID externe consiste à traiter et à prévenir le problème du député confus.

En AWS, l'usurpation d'identité interservices peut entraîner un problème de confusion chez les adjoints. L’usurpation d’identité entre services peut se produire lorsqu’un service (le service appelant) appelle un autre service (le service appelé). Le service appelant peut être manipulé pour utiliser ses autorisations afin d'agir sur les ressources d'un autre client de sorte qu'il n'y aurait pas accès autrement.

Prévention du problème de l'adjoint confus entre comptes

Le diagramme suivant illustre le problème de l'adjoint confus entre comptes.



Ce scénario suppose que :

- AWS 1 est le vôtre Compte AWS.
- AWS 1 : ExampleRole est un rôle dans votre compte. La politique d'approbation du rôle approuve Example Corp en désignant le compte AWS d'Example Corp comme celui qui peut endosser le rôle.

Voici ce qui se produit :

1. Lorsque vous commencez à utiliser le service d'Example Corp, vous fournissez l'ARN AWS 1 : ExampleRole à Example Corp.
2. Example Corp utilise cet ARN de rôle pour obtenir des informations d'identification de sécurité temporaires afin d'accéder aux ressources de votre Compte AWS. Ainsi, vous approuvez Example Corp en tant que « député » autorisé à agir pour votre compte.
3. Un autre AWS client commence également à utiliser le service d'Example Corp, et ce client fournit également l'ARN AWS 1 : ExampleRole for Example Corp à utiliser. On peut supposer que l'autre client a appris ou deviné le AWS 1 : ExampleRole, ce qui n'est pas un secret.
4. Lorsque l'autre client demande à Example Corp d'accéder aux AWS ressources (qu'il prétend être) de son compte, Example Corp utilise AWS 1 : ExampleRole pour accéder aux ressources de votre compte.

C'est ainsi que l'autre client peut obtenir un accès non autorisé à vos ressources. Du fait que cet autre client a été en mesure de tromper Example Corp pour agir involontairement sur vos ressources, Example Corp est à présent un « député confus ».

Exemple Corp peut résoudre le problème de l'adjoint confus en exigeant que vous incluiez la vérification de la condition `ExternalId` dans la politique d'approbation du rôle. Exemple Corp génère une valeur `ExternalId` unique pour chaque client et utilise cette valeur dans sa demande

de prise en charge du rôle. La valeur `ExternalId` doit être unique parmi les clients d'Example Corp et contrôlée par Example Corp, et non par ses clients. C'est pourquoi vous l'obtenez d'Example Corp et que vous ne créez pas le vôtre. Cela évite à Example Corp d'être un adjoint confus et d'accorder l'accès aux AWS ressources d'un autre compte.

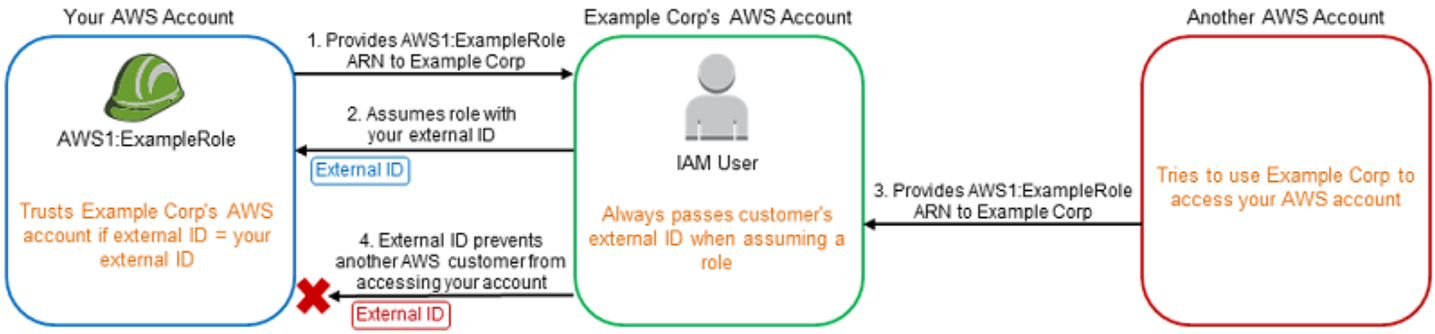
Dans notre scénario, imaginez que l'identifiant unique d'Example Corp pour vous soit 12345, et que son identifiant pour l'autre client soit 67890. Ces identifiants sont simplifiés pour ce scénario. En général, ces identifiants sont des GUID. Supposons que ces identifiants sont uniques pour chaque client d'Example Corp, ils constituent des valeurs sensibles à utiliser pour l'ID externe.

Example Corp vous attribue la valeur d'ID externe 12345. Vous devez ensuite ajouter un élément `Condition` à la politique d'approbation du rôle qui nécessite que la valeur de `sts:ExternalId` soit 12345, comme suit :

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "AWS": "Example Corp's AWS Account ID"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "sts:ExternalId": "12345"
      }
    }
  }
}
```

L'élément Condition de cette politique permet à Example Corp d'assumer le rôle uniquement lorsque l'appel d' AssumeRole API inclut la valeur d'ID externe 12345. Example Corp s'assure que chaque fois qu'elle assume un rôle au nom d'un client, elle inclut toujours la valeur d'identifiant externe de ce client dans l' AssumeRole appel. Même si un autre client fournit votre ARN à Example Corp, il ne peut pas contrôler l'ID externe qu'Example Corp inclut dans sa demande AWS. Cela permet d'éviter qu'un client non autorisé obtienne l'accès à vos ressources.

Le schéma suivant illustre ce processus.



1. Comme précédemment, lorsque vous commencez à utiliser le service d'Example Corp, vous fournissez l'ARN AWS 1 : ExampleRole à Example Corp.
2. Lorsque Example Corp utilise ce rôle ARN pour assumer le rôle AWS 1 : ExampleRole, Example Corp inclut votre ID externe (12345) dans l'appel d' AssumeRole API. L'ID externe correspond à la politique de confiance du rôle, de sorte que l'appel d' AssumeRole API aboutit et Example Corp obtient des informations de sécurité temporaires pour accéder aux ressources de votre. Compte AWS
3. Un autre AWS client commence également à utiliser le service d'Example Corp et, comme auparavant, ce client fournit également l'ARN AWS 1 : ExampleRole for Example Corp à utiliser.
4. Mais cette fois, lorsque Example Corp tente d'assumer le rôle AWS 1 : ExampleRole, elle fournit l'ID externe associé à l'autre client (67890). L'autre client est dans l'impossibilité de changer cela. Example Corp procède ainsi, car la demande pour utiliser le rôle provient de l'autre client, 67890 indique donc les circonstances dans lesquelles Example Corp agit. Comme vous avez ajouté une condition avec votre propre ID externe (12345) à la politique de confiance de AWS 1 : ExampleRole, l'appel d' AssumeRole API échoue. L'autre client se voit refuser l'autorisation d'accéder aux ressources de votre compte (indiqué par la croix « X » rouge dans le diagramme).

L'ID externe empêche tout autre client de forcer Example Corp de manière trompeuse à accéder involontairement à vos ressources.

Prévention du problème de l’adjoint confus entre services

Nous recommandons d'utiliser les clés de contexte de condition globale `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` ou `aws:SourceOrgPaths` dans les politiques basées sur les ressources pour limiter les autorisations dont dispose un service pour une ressource spécifique. `aws:SourceArn`À utiliser pour associer une seule ressource à un accès multiservice. `aws:SourceAccount`À utiliser pour associer n'importe quelle ressource de ce compte à l'utilisation interservices. `aws:SourceOrgID`À utiliser pour permettre à n'importe quelle ressource provenant

de n'importe quel compte au sein d'une organisation d'être associée à l'utilisation interservices. `aws:SourceOrgPaths`À utiliser pour associer toute ressource provenant de comptes situés dans un AWS Organizations chemin à l'utilisation interservices. Pour de plus amples informations sur l'utilisation et la compréhension des chemins, veuillez consulter Présentation du chemin d'entité AWS Organizations.

Le moyen le plus granulaire de se protéger contre le problème de l'adjoint confus est d'utiliser la clé de contexte de condition globale `aws:SourceArn` avec l'ARN complet de la ressource dans vos politiques basées sur les ressources. Si vous ne connaissez pas l'ARN complet de la ressource ou si vous spécifiez plusieurs ressources, utilisez la clé de contexte de condition globale `aws:SourceArn` avec des caractères génériques (*) pour les parties inconnues de l'ARN. Par exemple, `arn:aws:`*`servicename`*`:*:`*`123456789012`*`:*`.

Si la valeur `aws:SourceArn` ne contient pas l'ID du compte, tel qu'un ARN de compartiment Amazon S3, vous devez utiliser à la fois `aws:SourceAccount` et `aws:SourceArn` pour limiter les autorisations.

Pour se protéger contre le problème de l'adjoint confus à grande échelle, utilisez la clé de contexte de condition globale `aws:SourceOrgID` ou `aws:SourceOrgPaths` avec l'ID de l'organisation ou le chemin de l'organisation de la ressource dans vos politiques basées sur les ressources. Lorsque vous ajoutez, supprimez et déplacez des comptes dans votre organisation, les politiques qui contiennent la clé `aws:SourceOrgID` ou `aws:SourceOrgPaths` incluront automatiquement les bons comptes et vous n'avez pas besoin de mettre manuellement à jour les polices.

Pour les politiques de confiance en matière de non-service-linked rôle, chaque service inclus dans la politique de confiance a effectué l'`iam:PassRole`action nécessaire pour vérifier que le rôle se trouve sur le même compte que le service d'appel. Par conséquent, l'utilisation de `aws:SourceAccount`, `aws:SourceOrgID` ou `aws:SourceOrgPaths` avec ces politiques d'approbation n'est pas nécessaire. L'utilisation de `aws:SourceArn` dans une politique d'approbation vous permet de spécifier les ressources pour lesquelles un rôle peut être assumé en son nom, comme l'ARN d'une fonction Lambda. Certaines politiques Services AWS d'utilisation `aws:SourceAccount` et `aws:SourceArn` de confiance s'appliquent aux rôles nouvellement créés, mais l'utilisation des clés n'est pas obligatoire pour les rôles existants dans votre compte.

> **Note**
>
> Services AWS qui s'intègrent à AWS Key Management Service
> l'utilisation de l'attribution de clés KMS ne prennent pas en charge les clés

`aws:SourceArn`aws:SourceAccount,`aws:SourceOrgID`, ou les clés de `aws:SourceOrgPaths` condition. L'utilisation de ces clés de condition dans une politique de clé KMS entraînera un comportement inattendu si la clé est également utilisée par le Services AWS biais de l'attribution de clés KMS.

Prévention interservices confuse des adjoints pour AWS Security Token Service

De nombreux AWS services nécessitent que vous utilisiez des rôles pour permettre au service d'accéder aux ressources d'un autre service en votre nom. Un rôle qu'un service endosse pour effectuer des actions en votre nom s'appelle un rôle de service. Un rôle nécessite deux politiques : une politique d'approbation de rôle qui spécifie le principal autorisé à endosser le rôle et une politique d'autorisations qui spécifie ce qui peut être fait avec le rôle. Une politique d'approbation de rôle est le seul type de politique basée sur les ressources dans IAM. D'autres Services AWS ont des politiques basées sur les ressources, telles qu'une politique de compartiment Amazon S3.

Lorsqu'un service endosse un rôle en votre nom, le principal du service doit être autorisé à effectuer l'action `sts:AssumeRole` dans la politique d'approbation des rôles. Lorsqu'un service appelle`sts:AssumeRole`, AWS STS renvoie un ensemble d'informations d'identification de sécurité temporaires que le principal du service utilise pour accéder aux ressources autorisées par la politique d'autorisation du rôle. Lorsqu'un service endosse un rôle dans votre compte, vous pouvez inclure les clés de contexte de condition globale `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` ou `aws:SourceOrgPaths` dans votre politique d'approbation des rôles afin de limiter l'accès au rôle aux seules demandes générées par les ressources attendues.

Par exemple, dans AWS Systems Manager Incident Manager, vous devez choisir un rôle pour autoriser Incident Manager à exécuter un document d'automatisation de Systems Manager en votre nom. Le document d'automatisation peut inclure des plans de réponse automatisés pour les incidents déclenchés par des CloudWatch alarmes ou EventBridge des événements. Dans l'exemple de politique d'approbation de rôle suivant, vous pouvez utiliser la clé de condition `aws:SourceArn` pour limiter l'accès à la fonction du service sur la base de l'ARN de l'enregistrement d'incident. Seuls les enregistrements d'incidents qui sont créés à partir de la ressource `myresponseplan` du plan de réponse peuvent utiliser ce rôle.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
```

```
    "Principal": {
      "Service": "ssm-incidents.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:ssm-incidents:*:111122223333:incident-
record/myresponseplan/*"
      }
    }
  }
}
```

> **Note**
>
> Toutes les intégrations de services ne sont pas accompagnées `aws:SourceArn` de `aws:SourceAccount` clés de AWS STS support ou de `aws:SourceOrgPaths` condition. `aws:SourceOrgID` L'utilisation de ces clés dans les politiques d'approbation IAM avec des intégrations non prises en charge peut entraîner un comportement inattendu.

Octroyer l'accès à des utilisateurs authentifiés en externe (fédération d'identité)

Vos utilisateurs ont peut-être déjà une identité extérieure AWS, par exemple dans votre annuaire d'entreprise. Si ces utilisateurs ont besoin de travailler avec AWS des ressources (ou de travailler avec des applications qui accèdent à ces ressources), ils ont également besoin d'informations d'identification AWS de sécurité. A l'aide d'un rôle IAM, vous pouvez définir des autorisations pour des utilisateurs dont l'identité est fédérée par votre organisation ou un fournisseur d'identité (IdP) tiers.

> **Note**
>
> En tant que bonne pratique de sécurité, nous vous recommandons de gérer l'accès des utilisateurs dans IAM Identity Center à l'aide de la fédération d'identité plutôt que de créer des utilisateurs IAM. Pour en savoir plus sur les situations spécifiques dans lesquelles un utilisateur IAM est nécessaire, veuillez consulter Quand créer un utilisateur IAM (au lieu d'un rôle).

Fédération d'utilisateurs d'une application mobile ou web à l'aide d'Amazon Cognito

Si vous créez une application mobile ou Web qui accède aux AWS ressources, l'application a besoin d'informations d'identification de sécurité pour pouvoir envoyer des demandes programmatiques à. AWS Pour la plupart des scénarios impliquant des applications mobiles, nous recommandons d'utiliser Amazon Cognito. Vous pouvez utiliser ce service avec le SDK AWS mobile pour iOS et AWS le SDK mobile pour Android et Fire OS afin de créer des identités uniques pour les utilisateurs et de les authentifier afin de sécuriser l'accès à vos ressources. AWS Amazon Cognito prend en charge les mêmes fournisseurs d'identité que ceux répertoriés dans la section suivante, ainsi que les identités authentifiées par le développeur et les accès non authentifiés (invité). Amazon Cognito fournit également des opérations d'API pour la synchronisation des données utilisateur afin de les conserver à mesure que les utilisateurs passent d'un périphérique à l'autre. Pour plus d'informations, veuillez consulter Utilisation d'Amazon Cognito pour les applications mobiles.

Fédération d'utilisateurs à l'aide de fournisseurs d'identité publics ou d'OpenID Connect

Dans la mesure du possible, utilisez Amazon Cognito pour les scénarios d'applications mobiles et Web. Amazon Cognito effectue la majeure partie du behind-the-scenes travail avec les services des fournisseurs d'identité publics pour vous. Il fonctionne avec les mêmes services tiers et prend également en charge les connexions anonymes. Toutefois, dans le cas de scénarios plus complexes, vous pouvez avoir directement recours à un service tiers tel que Login with Amazon, Facebook, Google, ou tout autre IdP compatible avec OpenID Connect (OIDC). Pour plus d'informations sur l'utilisation de la fédération OIDC à l'aide de l'un de ces services, consultezFédération OIDC.

Fédération d'utilisateurs avec SAML 2.0

Si votre organisation utilise déjà un progiciel de fournisseur d'identité compatible avec le SAML 2.0 (Security Assertion Markup Language 2.0), vous pouvez créer un climat de confiance entre votre organisation en tant que fournisseur d'identité (IdP) et en AWS tant que fournisseur de services. Vous pouvez ensuite utiliser SAML pour fournir à vos utilisateurs une authentification unique (SSO) fédérée AWS Management Console ou un accès fédéré aux opérations d'API d'appel. AWS Par exemple, si votre entreprise utilise Microsoft Active Directory et Active Directory Federation Services, vous pouvez utiliser la fédération à l'aide de SAML 2.0. Pour plus d'informations sur l'utilisation des utilisateurs fédérés avec SAML 2.0, consultez Fédération SAML 2.0.

Fédération d'utilisateurs via la création d'une application de broker d'identité personnalisé

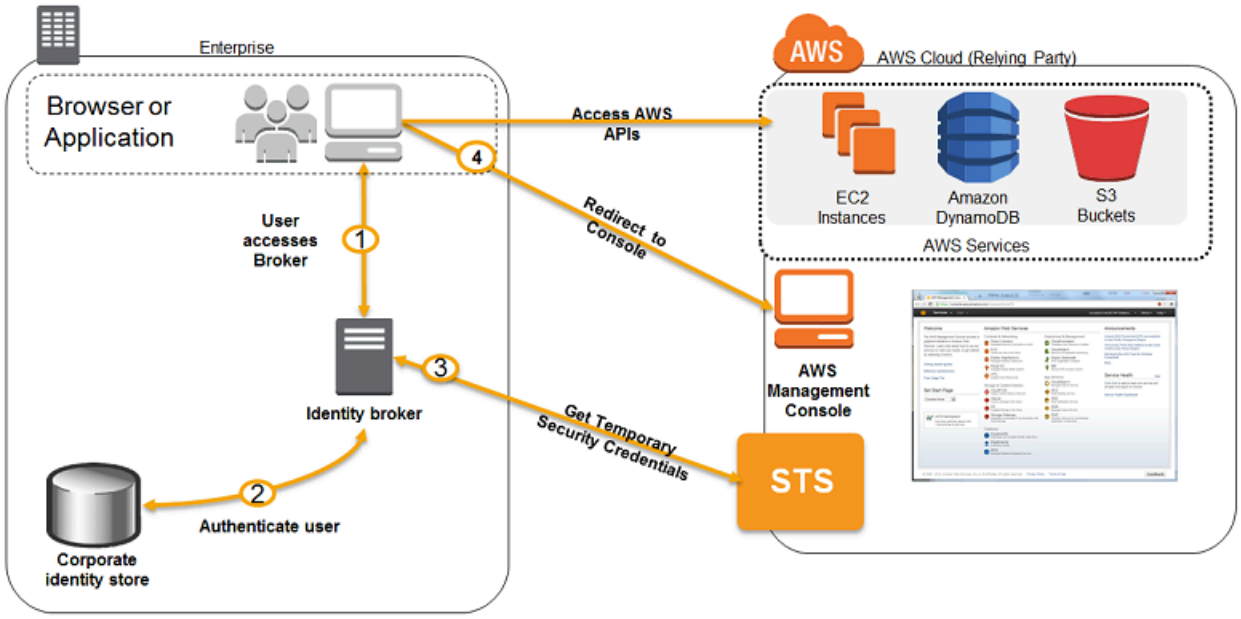
Si votre base d'identités n'est pas compatible avec SAML 2.0, vous pouvez créer une application de broker d'identité personnalisé capable d'exécuter une fonction similaire. L'application broker

authentifie les utilisateurs, leur demande des informations d'identification temporaires AWS, puis les fournit à l'utilisateur pour qu'il accède aux AWS ressources.

Par exemple, de nombreux employés d'Example Corp. doivent exécuter des applications internes qui accèdent aux AWS ressources de l'entreprise. Ils disposent déjà d'identités dans le système d'identité et d'authentification de l'entreprise et, par conséquent, Example Corp. ne souhaite pas créer un utilisateur IAM séparé pour chacun de ses employés.

Bob est développeur chez Example Corp. Pour permettre aux applications internes d'Example Corp. d'accéder aux AWS ressources de l'entreprise, Bob développe une application personnalisée de courtier d'identité. L'application vérifie que les employés sont connectés au système d'identité et d'authentification existant d'Example Corp., par exemple LDAP, Active Directory ou un autre système. L'application de broker d'identité obtient ensuite des informations d'identification de sécurité temporaires pour les employés. Ce scénario est similaire au précédent (une application mobile utilisant un système d'authentification personnalisé), sauf que les applications qui ont besoin d'accéder aux AWS ressources s'exécutent toutes sur le réseau de l'entreprise et que l'entreprise dispose d'un système d'authentification existant.

Pour obtenir les informations d'identification de sécurité temporaires, l'application de broker d'identité appelle `AssumeRole` ou `GetFederationToken`, selon la façon dont Bob veut gérer les politiques des utilisateurs et le délai d'expiration des informations d'identification. Pour plus d'informations sur les différences entre ces opérations d'API, consultez Informations d'identification de sécurité temporaires dans IAM et Contrôle des autorisations affectées aux informations d'identification de sécurité temporaires.) L'appel renvoie des informations de sécurité temporaires composées d'un identifiant de clé d' AWS accès, d'une clé d'accès secrète et d'un jeton de session. L'application de broker d'identité rend ensuite ces informations d'identification de sécurité temporaires accessibles à l'application interne de l'entreprise. L'application peut alors utiliser ces informations d'identification de sécurité temporaires pour appeler directement AWS . L'application met en cache les informations d'identification jusqu'à ce qu'elles parviennent à expiration, puis demande un nouvel ensemble d'informations d'identification temporaires. L'illustration suivante décrit ce scénario.



Ce scénario utilise les attributs suivants :

- L'application de broker d'identité dispose d'autorisations d'accès à l'API du service de jeton (STS) d'IAM pour créer des informations d'identification de sécurité temporaires.
- L'application de broker d'identité peut vérifier que les employés sont authentifiés dans le système d'authentification existant.
- Les utilisateurs peuvent obtenir une URL temporaire qui leur donne accès à la console de AWS gestion (appelée authentification unique).

Pour plus d'informations sur la création d'informations d'identification de sécurité temporaires, consultez Demande d'informations d'identification temporaires de sécurité. Pour plus d'informations sur l'accès des utilisateurs fédérés à la console AWS de gestion, consultezPermettre aux utilisateurs fédérés SAML 2.0 d'accéder au AWS Management Console.

Utilisation des rôles liés à un service

Un rôle lié à un service est un type unique de rôle IAM directement lié à un service AWS . Les rôles liés au service sont prédéfinis par le service et incluent toutes les autorisations dont le service a besoin pour appeler d'autres AWS services en votre nom. Le service lié définit aussi la manière dont vous créez, modifiez et supprimez un rôle lié à un service. Un service peut créer ou supprimer automatiquement le rôle. Il peut vous permettre de créer, modifier ou supprimer le rôle dans le cadre des opérations d'un assistant ou d'un processus du service. Ou il peut vous demander

d'utiliser IAM pour créer ou supprimer le rôle. Quelle que soit la méthode, les rôles liés à un service simplifient le processus de configuration d'un service étant donné que vous n'avez pas besoin d'ajouter manuellement les autorisations requises pour que le service effectue des actions en votre nom.

> **Note**
>
> N'oubliez pas que les fonctions du service sont différentes des rôles liés à un service. Une fonction de service est un rôle IAM qu'un service endosse pour accomplir des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer une fonction du service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM. Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés à un service apparaissent dans votre Compte AWS fichier et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

Le service lié définit les autorisations de ses rôles liés à un service et, sauf définition contraire, seul ce service peut endosser les rôles. Les autorisations définies comprennent la politique d'approbation et la politique d'autorisation. De plus, cette politique d'autorisation ne peut pas être attachée à une autre entité IAM.

Avant que vous ne puissiez supprimer les rôles, vous devez d'abord supprimer les ressources qui leur sont associées. Vos ressources sont ainsi protégées, car vous ne pouvez pas involontairement supprimer l'autorisation d'accéder aux ressources.

> **Tip**
>
> Pour plus d'informations concernant la prise en charge par les services des rôles liés à un service, consultez AWS services qui fonctionnent avec IAM et recherchez les services affichant Oui dans la colonne Rôle lié à un service. Choisissez un Yes (Oui) ayant un lien permettant de consulter les détails du rôle pour ce service.

Autorisations de rôles liés à un service

Vous devez configurer les autorisations d'une entité IAM (utilisateur ou rôle) de manière à permettre à l'utilisateur ou au rôle de créer ou modifier le rôle lié à un service.

> **Note**
>
> L'ARN d'un rôle lié à un service comprend un principal de service indiqué dans les stratégies ci-dessous comme *SERVICE-NAME*.amazonaws.com. N'essayez pas de deviner le principal du service, car il distingue les majuscules et minuscules et le format peut varier d'un AWS service à l'autre. Pour afficher le principal de service d'un service, consultez la documentation de son rôle lié à un service.

Pour permettre à une entité IAM de créer un rôle spécifique lié à un service

Ajoutez la politique suivante à l'entité IAM qui doit créer le rôle lié à un service.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*",
            "Condition": {"StringLike": {"iam:AWSServiceName": "SERVICE-
NAME.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
        }
    ]
}
```

Pour permettre à une entité IAM de créer un rôle lié à un service

Ajoutez l'instruction suivante à la politique d'autorisation de l'entité IAM qui doit créer un rôle lié à un service, ou un rôle de service incluant les politiques requises. Cette instruction de politique n'autorise pas l'entité IAM à attacher une politique à ce rôle.

```
{
    "Effect": "Allow",
    "Action": "iam:CreateServiceLinkedRole",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Pour permettre à une entité IAM de modifier la description de rôles liés à un service

Ajoutez l'instruction suivante à la politique d'autorisation de l'entité IAM qui doit modifier la description d'un rôle lié à un service ou d'un rôle de service.

```
{
    "Effect": "Allow",
    "Action": "iam:UpdateRoleDescription",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Pour permettre à une entité IAM de supprimer un rôle spécifique lié à un service

Ajoutez l'instruction suivante à la politique d'autorisation de l'entité IAM qui doit supprimer le rôle lié à un service.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
}
```

Pour permettre à une entité IAM de supprimer un rôle lié à un service

Ajoutez l'instruction suivante à la politique d'autorisation de l'entité IAM qui doit supprimer un rôle lié à un service, mais pas le rôle de service.

```
{
```

```
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Pour permettre à une entité IAM de transmettre un rôle existant au service

Certains AWS services vous permettent de transmettre un rôle existant au service, au lieu de créer un nouveau rôle lié au service. Pour ce faire, un utilisateur doit disposer des autorisations nécessaires pour transmettre le rôle au service. Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit transmettre un rôle. Cette instruction de politique autorise également l'entité à afficher une liste des rôles à partir de laquelle ils peuvent choisir le rôle à transmettre. Pour plus d'informations, consultez Octroi d'autorisations à un utilisateur pour transférer un rôle à un service AWS.

```
{
  "Sid": "PolicyStatementToAllowUserToListRoles",
  "Effect": "Allow",
  "Action": ["iam:ListRoles"],
  "Resource": "*"
},
{
  "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
  "Effect": "Allow",
  "Action": [ "iam:PassRole" ],
  "Resource": "arn:aws:iam::account-id:role/my-role-for-XYZ"
}
```

Autorisations indirectes avec rôles liés à un service

Les autorisations accordées par un rôle lié à un service peuvent être transférées indirectement à d'autres utilisateurs et rôles. Lorsqu'un rôle lié à un service est utilisé par un AWS service, ce rôle lié au service peut utiliser ses propres autorisations pour appeler d'autres services. AWS Cela signifie que les utilisateurs et les rôles ayant l'autorisation d'appeler un service qui utilise un rôle lié à un service peuvent avoir un accès indirect aux services auxquels ce rôle lié à un service peut accéder.

Par exemple, lorsque vous créez une instance de base données Amazon RDS, un rôle lié à un service pour RDS est automatiquement créé s'il n'existe pas déjà. Ce rôle lié au service permet à

RDS d'appeler Amazon EC2, Amazon SNS, Amazon CloudWatch Logs et Amazon Kinesis en votre nom. Si vous autorisez les utilisateurs et les rôles de votre compte à modifier ou à créer des bases de données RDS, ils pourront peut-être interagir indirectement avec Amazon EC2, Amazon SNS, les journaux Amazon Logs et les ressources CloudWatch Amazon Kinesis en appelant RDS, car RDS utiliserait son rôle lié au service pour accéder à ces ressources.

Création d'un rôle lié à un service

La méthode que vous utilisez pour créer un rôle lié à un service dépend dudit service. Dans certains cas, vous n'avez pas besoin de créer manuellement un rôle lié à un service. Par exemple, lorsque vous terminez une action donnée (comme créer une ressource) dans le service, le service peut créer le rôle lié au service à votre place. Ou, si vous utilisiez un service avant qu'il ne prenne en charge les rôles liés à un service, alors le service peut avoir créé automatiquement le rôle dans votre compte. Pour en savoir plus, veuillez consulter la section Un nouveau rôle est apparu dans mon compte AWS.

Dans d'autres cas, le service peut prendre en charge la création manuelle d'un rôle lié à un service à l'aide la console, l'API ou de la CLI. Pour plus d'informations concernant la prise en charge par les services des rôles liés à un service, consultez AWS services qui fonctionnent avec IAM et recherchez les services affichant Oui dans la colonne Rôle lié à un service. Pour savoir si le service prend en charge la création du rôle lié à un service, choisissez le lien Oui pour afficher la documentation du rôle lié à ce service.

Si le service ne prend pas en charge la création du rôle, alors vous pouvez utiliser IAM pour créer le rôle lié à un service.

> **Important**
>
> Les rôles liés à un service comptent dans votre limite de rôles IAM dans un Compte AWS, mais si vous avez atteint cette limite, vous pouvez toujours créer des rôles liés à un service dans votre compte. Seuls les rôles liés à un service peuvent dépasser la limite sans conséquence.

Création d'un rôle lié à un service (console)

Avant de créer un rôle lié à un service dans IAM, sachez si le service lié crée automatiquement des rôles liés au service et si vous pouvez créer le rôle depuis la console, l'API ou la CLI du service.

Pour créer un rôle lié à un service (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles). Puis, choisissez Create role (Créer un rôle).
3. Choisissez le type de fonction du service AWS .
4. Choisissez le cas d'utilisation de votre service. Les cas d'utilisation sont définis par le service pour inclure la politique d'approbation requise par le service. Ensuite, choisissez Suivant.
5. Choisissez une ou plusieurs politiques d'autorisation à attacher au rôle. En fonction du cas d'utilisation sélectionné, le service peut effectuer n'importe laquelle des options suivantes :

   - Définissez les autorisations utilisées par le rôle.
   - Vous permet de choisir parmi un ensemble limité d'autorisations.
   - Vous permet de choisir parmi toutes les autorisations.
   - Vous permet de choisir de ne sélectionner aucune stratégie pour le moment, mais de créer les politiques plus tard et de les attacher au rôle.

   Cochez la case en regard de la politique qui affecte les autorisations que vous voulez octroyer au rôle, puis choisissez Next (Suivant).

   > **Note**
   >
   > Les autorisations que vous spécifiez sont disponibles à toutes les entités qui utilisent le rôle. Par défaut, un rôle ne dispose d'aucune autorisation.

6. Pour Nom du rôle, le degré de la personnalisation du nom du rôle est défini par le service. Si le service définit le nom du rôle, alors cette option n'est pas modifiable. Dans d'autres cas, le service peut définir un préfixe pour le rôle ou vous laisser saisir un suffixe facultatif.

   Si possible, saisissez un suffixe de nom de rôle à ajouter au nom par défaut. Ce suffixe vous aide à identifier l'objectif de ce rôle. Les noms de rôle de votre compte AWS doivent être uniques. Ils ne sont pas sensibles à la casse. Par exemple, vous ne pouvez pas créer deux rôles nommés **`<service-linked-role-name>_SAMPLE`** et **`<service-linked-role-name>_sample`**. Différentes entités peuvent référencer le rôle et il n'est donc pas possible de modifier son nom après sa création.

7. (Facultatif) Dans le champ Description, modifiez la description du nouveau rôle lié à un service.
8. Vous ne pouvez pas attacher des balises à des rôles liés à un service lors de la création. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Balisage des ressources IAM.
9. Passez en revue les informations du rôle, puis choisissez Créer un rôle.

Création d'un rôle lié à un service (AWS CLI)

Avant de créer un rôle lié à un service dans IAM, sachez si le service lié crée automatiquement des rôles liés au service et si vous pouvez créer le rôle depuis la CLI du service. Si la CLI du service n'est pas prise en charge, vous pouvez employer les commandes IAM pour créer un rôle lié à un service avec la politique d'approbation et les politiques en ligne dont le service a besoin pour endosser le rôle.

Pour créer un rôle lié à un service (AWS CLI)

Exécutez la commande suivante :

```
aws iam create-service-linked-role --aws-service-name SERVICE-NAME.amazonaws.com
```

Création d'un rôle lié à un service (API AWS)

Avant de créer un rôle lié à un service dans IAM, sachez si le service lié crée automatiquement des rôles liés au service et si vous pouvez créer le rôle depuis l'API du service. Si l'API de service n'est pas prise en charge, vous pouvez utiliser l' AWS API pour créer un rôle lié au service avec la politique de confiance et les politiques intégrées dont le service a besoin pour assumer ce rôle.

Pour créer un rôle lié à un service (API)AWS

Utilisez l'appel d'API CreateServiceLinkedRole. Dans la demande, spécifiez un nom de service sous la forme *SERVICE_NAME_URL*`.amazonaws.com`.

Par exemple, pour créer le rôle lié à un service Robots Lex, utilisez `lex.amazonaws.com`.

Modification d'un rôle lié à un service

La méthode que vous utilisez pour modifier un rôle lié à un service dépend dudit service. Certains services peuvent vous permettre de modifier les autorisations d'un rôle lié à un service depuis la console, l'API ou la CLI. Cependant, une fois un rôle lié à un service créé, vous ne pouvez pas

changer le nom d'un rôle car plusieurs entités peuvent faire référence au rôle. Vous pouvez modifier la description de n'importe quel rôle depuis la console IAM, l'API ou la CLI.

Pour plus d'informations concernant la prise en charge par les services des rôles liés à un service, consultez AWS services qui fonctionnent avec IAM et recherchez les services affichant Oui dans la colonne Rôle lié à un service. Pour savoir si le service prend en charge la modification du rôle lié à un service, choisissez le lien Oui pour afficher la documentation du rôle lié à ce service.

Modification de la description d'un rôle lié à un service (console)

Vous pouvez utiliser la console IAM pour modifier la description d'un rôle lié à un service.

Pour modifier la description d'un rôle lié à un service (console)

1. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles).
2. Choisissez le nom du rôle à modifier.
3. A l'extrême droite de Description du rôle, choisissez Edit (Modifier).
4. Saisissez une nouvelle description dans la zone et choisissez Save (Enregistrer).

Modification de la description d'un rôle lié à un service (AWS CLI)

Vous pouvez utiliser les commandes IAM depuis le AWS CLI pour modifier la description d'un rôle lié à un service.

Pour changer la description d'un rôle lié à un service (AWS CLI)

1. (Facultatif) Pour afficher la description actuelle d'un rôle, exécutez les commandes suivantes :

   ```
   aws iam get-role --role-name ROLE-NAME
   ```

   Utilisez le nom du rôle, pas l'ARN, pour faire référence aux rôles avec les commandes CLI. Par exemple, si un rôle a l'ARN : `arn:aws:iam::123456789012:role/myrole`, vous faites référence au rôle en tant que **`myrole`**.
2. Pour mettre à jour la description d'un rôle lié à un service, exécutez la commande suivante :

   ```
   aws iam update-role --role-name ROLE-NAME --description OPTIONAL-DESCRIPTION
   ```

Modification d'une description de rôle liée à un service (API)AWS

Vous pouvez utiliser l' AWS API pour modifier la description d'un rôle lié à un service.

Pour modifier la description d'un rôle lié à un service (API)AWS

1. (Facultatif) Pour afficher la description actuelle d'un rôle, appelez l'opération suivante et spécifiez le nom du rôle :

   AWS API : GetRole
2. Pour mettre à jour la description d'un rôle, appelez l'opération suivante et spécifiez le nom (et la description facultative) du rôle :

   AWS API : UpdateRole

Suppression d'un rôle lié à un service

La méthode que vous utilisez pour créer un rôle lié à un service dépend dudit service. Dans certains cas, vous n'avez pas besoin de supprimer manuellement un rôle lié à un service. Par exemple, lorsque vous terminez une action donnée (comme supprimer une ressource) dans le service, le service peut supprimer le rôle lié au service à votre place.

Dans d'autres cas, le service peut prendre en charge la suppression manuelle d'un rôle lié à un service à partir de la console, de l'API ou de la AWS CLI.

Pour plus d'informations concernant la prise en charge par les services des rôles liés à un service, consultez AWS services qui fonctionnent avec IAM et recherchez les services affichant Oui dans la colonne Rôle lié à un service. Pour savoir si le service prend en charge la suppression du rôle lié à un service, choisissez le lien Oui pour afficher la documentation du rôle lié à ce service.

Si le service ne prend pas en charge la suppression du rôle, vous pouvez supprimer le rôle lié au service depuis la console IAM, l'API ou. AWS CLI Si vous n’avez plus besoin d’utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n’avez aucune entité inutilisée qui n’est pas surveillée ou gérée activement. Cependant, vous devez nettoyer votre rôle lié à un service avant de pouvoir le supprimer.

Nettoyage d'un rôle lié à un service

Avant de pouvoir utiliser IAM pour supprimer un rôle lié à un service, vous devez d'abord vérifier qu'aucune session n'est active pour le rôle et supprimer toutes les ressources utilisées par le rôle.

Pour vérifier si une session est active pour le rôle lié à un service dans la console IAM

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles). Choisissez ensuite le nom (et non pas la case à cocher correspondante) du rôle lié à un service.
3. Sur la page Récapitulatif du rôle sélectionné, choisissez l'onglet Access Advisor.
4. Dans l'onglet Access Advisor, consultez l'activité récente pour le rôle lié à un service.

> **Note**
>
> Si vous n'êtes pas certain que le service utilise le rôle lié à un service, vous pouvez essayer de supprimer le rôle. Si le service utilise le rôle, la suppression échoue et vous avez accès aux régions dans lesquelles le rôle est utilisé. Si le rôle est utilisé, vous devez attendre que la session se termine avant de pouvoir le supprimer. Vous ne pouvez pas révoquer la session d'un rôle lié à un service.

Pour supprimer des ressources utilisées par un rôle lié à un service

Pour plus d'informations concernant la prise en charge par les services des rôles liés à un service, consultez AWS services qui fonctionnent avec IAM et recherchez les services affichant Oui dans la colonne Rôle lié à un service. Pour savoir si le service prend en charge la suppression du rôle lié à un service, choisissez le lien Oui pour afficher la documentation du rôle lié à ce service. Consultez la documentation relative à ce service pour savoir comment supprimer des ressources utilisées par votre rôle lié à un service.

Suppression d'un rôle lié à un service (console)

Vous pouvez utiliser la console IAM pour supprimer un rôle lié à un service.

Pour supprimer un rôle lié à un service (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles). Cochez ensuite la case en regard du nom du rôle que vous souhaitez supprimer, sans sélectionner le nom ou la ligne.

3. Pour Role actions (Actions de rôle) en haut de la page, sélectionnez Delete (Supprimer).
4. Dans la boîte de dialogue de confirmation, passez en revue les dernières informations consultées, qui indiquent la date à laquelle chacun des rôles sélectionnés a accédé pour la dernière fois à un AWS service. Cela vous permet de confirmer si le rôle est actif actuellement. Si vous souhaitez continuer, sélectionnez Oui, supprimer pour lancer la tâche de suppression du rôle.
5. Consultez les notifications de la console IAM pour surveiller la progression de la suppression du rôle lié à un service. Dans la mesure où la suppression du rôle lié à un service IAM est asynchrone, une fois que vous soumettez le rôle afin qu'il soit supprimé, la suppression peut réussir ou échouer.

   - Si la tâche réussit, le rôle est supprimé de la liste et une notification de succès s'affiche en haut de la page.
   - Si la tâche échoue, vous pouvez choisir View details (Afficher les détails) ou View Resources (Afficher les ressources) à partir des notifications pour connaître le motif de l'échec de la suppression. Si la suppression échoue car le rôle utilise les ressources du service, alors la notification comprend une liste de ressources, à condition que le service renvoie ces informations. Vous pouvez alors nettoyer les ressources et lancer à nouveau la tâche de suppression.

     > **Note**
     >
     > Vous devrez peut-être répéter ce processus plusieurs fois, en fonction des informations renvoyées par le service. Par exemple, il est possible que votre rôle lié à un service utilise six ressources et que votre service renvoie des informations sur cinq d'entre elles. Si vous nettoyez les cinq ressources et lancez à nouveau la tâche de suppression pour le rôle, la suppression échoue et le service indique la ressource restante. Un service peut renvoyer toutes les ressources, quelques ressources ou n'indiquer aucune ressource.

   - Si la tâche échoue et que la notification n'inclut pas de liste des ressources, le service peut ne pas renvoyer cette information. Pour savoir comment nettoyer les ressources pour ce service, consultez AWS services qui fonctionnent avec IAM. Identifiez votre service dans le tableau, puis choisissez le lien Yes (Oui) pour afficher la documentation relative au rôle lié à un service pour ce service.

Suppression d'un rôle lié à un service (AWS CLI)

Vous pouvez utiliser les commandes IAM depuis le AWS CLI pour supprimer un rôle lié à un service.

Pour supprimer un rôle lié à un service (AWS CLI)

1. Si vous connaissez le nom du rôle lié à un service que vous souhaitez supprimer, saisissez la commande suivante pour répertorier le rôle dans votre compte :

   ```
   aws iam get-role --role-name role-name
   ```

   Utilisez le nom du rôle, pas l'ARN, pour faire référence aux rôles avec les commandes CLI. Par exemple, si un rôle a l'ARN : `arn:aws:iam::123456789012:role/myrole`, vous faites référence au rôle en tant que **`myrole`**.

2. Dans la mesure où un rôle lié à un service ne peut pas être supprimé s'il est utilisé ou si des ressources lui sont associées, vous devez envoyer une demande de suppression. Cette demande peut être refusée si ces conditions ne sont pas satisfaites. Vous devez capturer le `deletion-task-id` de la réponse afin de vérifier l'état de la tâche de suppression. Saisissez la commande suivante pour envoyer une demande de suppression d'un rôle lié à un service :

   ```
   aws iam delete-service-linked-role --role-name role-name
   ```

3. Saisissez la commande suivante pour vérifier l'état de la tâche de suppression :

   ```
   aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
   ```

   L'état de la tâche de suppression peut être `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED` ou `FAILED`. Si la suppression échoue, l'appel renvoie le motif de l'échec, afin que vous puissiez apporter une solution. Si la suppression échoue car le rôle utilise les ressources du service, alors la notification comprend une liste de ressources, à condition que le service renvoie ces informations. Vous pouvez alors nettoyer les ressources et lancer à nouveau la tâche de suppression.

   > **Note**
   >
   > Vous devrez peut-être répéter ce processus plusieurs fois, en fonction des informations renvoyées par le service. Par exemple, il est possible que votre rôle lié à un service utilise six ressources et que votre service renvoie des informations sur cinq d'entre

elles. Si vous nettoyez les cinq ressources et lancez à nouveau la tâche de suppression pour le rôle, la suppression échoue et le service indique la ressource restante. Un service peut renvoyer toutes les ressources, quelques ressources ou n'indiquer aucune ressource. Pour apprendre à nettoyer les ressources pour un service qui n'indique aucune ressource, consultez AWS services qui fonctionnent avec IAM. Identifiez votre service dans le tableau, puis choisissez le lien Yes (Oui) pour afficher la documentation relative au rôle lié à un service pour ce service.

Suppression d'un rôle lié à un service (API AWS)

Vous pouvez utiliser l' AWS API pour supprimer un rôle lié à un service.

Pour supprimer un rôle lié à un service (API)AWS

1. Pour soumettre une demande de suppression pour un rôle lié à un service, appelez. DeleteServiceLinkedRole Dans la demande, spécifiez le nom d'un rôle.

   Dans la mesure où un rôle lié à un service ne peut pas être supprimé s'il est utilisé ou si des ressources lui sont associées, vous devez envoyer une demande de suppression. Cette demande peut être refusée si ces conditions ne sont pas satisfaites. Vous devez capturer le `DeletionTaskId` de la réponse afin de vérifier l'état de la tâche de suppression.
2. Pour vérifier l'état de la suppression, appelez GetServiceLinkedRoleDeletionStatus. Dans la demande, spécifiez le `DeletionTaskId`.

   L'état de la tâche de suppression peut être `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED` ou `FAILED`. Si la suppression échoue, l'appel renvoie le motif de l'échec, afin que vous puissiez apporter une solution. Si la suppression échoue car le rôle utilise les ressources du service, alors la notification comprend une liste de ressources, à condition que le service renvoie ces informations. Vous pouvez alors nettoyer les ressources et lancer à nouveau la tâche de suppression.

   > **Note**
   >
   > Vous devrez peut-être répéter ce processus plusieurs fois, en fonction des informations renvoyées par le service. Par exemple, il est possible que votre rôle lié à un service utilise six ressources et que votre service renvoie des informations sur cinq d'entre elles. Si vous nettoyez les cinq ressources et lancez à nouveau la tâche de suppression

pour le rôle, la suppression échoue et le service indique la ressource restante. Un service peut renvoyer toutes les ressources, quelques ressources ou n'indiquer aucune ressource. Pour apprendre à nettoyer les ressources pour un service qui n'indique aucune ressource, consultez AWS services qui fonctionnent avec IAM. Identifiez votre service dans le tableau, puis choisissez le lien Yes (Oui) pour afficher la documentation relative au rôle lié à un service pour ce service.

Création de rôles IAM

Pour créer un rôle, vous pouvez utiliser les AWS Management Console AWS CLI outils pour Windows ou PowerShell l'API IAM.

Si vous utilisez le AWS Management Console, un assistant vous guide tout au long des étapes de création d'un rôle. Les étapes de l'assistant varient légèrement selon que vous créez un rôle pour un AWS service, pour un Compte AWS utilisateur fédéré ou pour un utilisateur fédéré.

Rubriques

- Création d'un rôle pour la délégation d'autorisations à un utilisateur IAM.
- Création d'un rôle pour la délégation d'autorisations à un service AWS
- Création d'un rôle pour un fournisseur d'identité tiers (fédération)
- Création d'un rôle à l'aide de politiques d'approbation personnalisées (console)
- Exemples de politiques pour la délégation d'accès

Création d'un rôle pour la délégation d'autorisations à un utilisateur IAM.

Vous pouvez utiliser les rôles IAM pour déléguer l'accès à vos AWS ressources. Avec les rôles IAM, vous pouvez établir des relations de confiance entre votre compte de confiance et d'autres comptes de AWS confiance. Le compte d'approbation est propriétaire des ressources auxquelles les utilisateurs ont accès , tandis que le compte approuvé contient les utilisateurs devant accéder aux ressources. Toutefois, il est possible qu'un autre compte détienne une ressource dans votre compte. Par exemple, le compte de confiance peut autoriser le compte approuvé à créer de nouvelles ressources, telles que la création de nouveaux objets dans un compartiment Amazon S3. Dans ce cas, le compte qui crée la ressource la détient et contrôle les personnes pouvant y accéder.

Une fois que vous avez créé la relation de confiance, un utilisateur IAM ou une application du compte sécurisé peut utiliser l'opération d'`AssumeRole`API AWS Security Token Service (AWS STS).

Cette opération fournit des informations de sécurité temporaires qui permettent d'accéder aux AWS ressources de votre compte.

Les deux comptes peuvent être contrôlés par vous-même, ou le compte contenant les utilisateurs peut être contrôlé par un tiers. Si l'autre compte associé aux utilisateurs est un compte Compte AWS que vous ne contrôlez pas, vous pouvez utiliser l'`externalId`attribut. L'ID externe peut être n'importe quel mot ou nombre convenu avec l'administrateur du compte tiers. Cette option ajoute automatiquement une condition à la politique d'approbation. Cette condition permet à l'utilisateur d'endosser le rôle uniquement si la demande inclut l'élément approprié `sts:ExternalID`. Pour plus d'informations, veuillez consulter Comment utiliser un identifiant externe lorsque vous accordez l'accès à vos AWS ressources à un tiers.

Pour plus d'informations sur la façon d'utiliser les rôles pour déléguer des autorisations, consultez Termes et concepts relatifs aux rôles. Pour en savoir plus sur l'utilisation d'un rôle de service afin de permettre aux services l'accès aux ressources de votre compte, consultez Création d'un rôle pour la délégation d'autorisations à un service AWS.

Création d'un rôle IAM (console)

Vous pouvez utiliser le AWS Management Console pour créer un rôle qu'un utilisateur IAM peut assumer. Supposons, par exemple, que votre organisation en dispose Comptes AWS de plusieurs pour isoler un environnement de développement d'un environnement de production. Pour des informations générales sur la création d'un rôle autorisant les utilisateurs du compte de développement à accéder aux ressources du compte de production, consultez Exemple de scénario utilisant des comptes de développement et de production distincts.

Pour créer un rôle (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console, choisissez Rôles, puis Créer un rôle.
3. Choisissez le type du rôle de l'Compte AWS.
4. Pour créer un rôle pour votre compte, choisissez This account (Ce compte). Pour créer un rôle pour un autre compte, choisissez Another Compte AWS (Autre) et saisissez l'Account ID (ID de compte) auquel vous voulez octroyer l'accès à vos ressources.

   L'administrateur du compte spécifié peut accorder l'autorisation d'endosser ce rôle à n'importe quel utilisateur IAM de ce compte. Pour ce faire, l'administrateur attache une politique à

l'utilisateur ou à un groupe qui donne l'autorisation pour l'action `sts:AssumeRole`. Cette politique doit spécifier l'ARN du rôle comme `Resource`.

5. Si vous accordez des autorisations aux utilisateurs d'un compte que vous ne contrôlez pas et si les utilisateurs ont l'intention d'assumer ce rôle par programmation, alors sélectionnez Require external ID (Demander un ID externe). L'ID externe peut être n'importe quel mot ou nombre convenu avec l'administrateur du compte tiers. Cette option ajoute automatiquement une condition à la politique d'approbation. Cette condition permet à l'utilisateur d'endosser le rôle uniquement si la demande inclut l'élément approprié `sts:ExternalID`. Pour plus d'informations, consultez Comment utiliser un identifiant externe lorsque vous accordez l'accès à vos AWS ressources à un tiers.

   > **Important**
   >
   > Le choix de cette option restreint l'accès au rôle uniquement par le biais AWS CLI des outils pour Windows PowerShell ou de l' AWS API. Cela est dû au fait que vous ne pouvez pas utiliser la AWS console pour passer à un rôle dont la politique de confiance est assortie d'une `externalId` condition. Vous pouvez néanmoins créer ce type d'accès par programmation, en écrivant un script ou une application à l'aide du kit SDK approprié. Pour plus d'informations et un exemple de script, consultez la section Comment activer l'accès multicompte AWS Management Console au blog sur la AWS sécurité.

6. Si vous souhaitez restreindre le rôle aux utilisateurs qui se connectent via une authentification MFA, sélectionnez Demander l'authentification MFA. Cela ajoute une condition à la politique d'approbation du rôle qui exige une authentification MFA. Un utilisateur qui veut endosser le rôle doit se connecter avec un mot de passe unique temporaire à partir d'un dispositif MFA configuré. Sans authentification MFA, les utilisateurs ne peuvent pas endosser le rôle. Pour plus d'informations sur l'authentification MFA, consultez Utilisation de l'authentification multifactorielle (MFA) dans AWS
7. Choisissez Suivant.
8. IAM inclut une liste des politiques AWS gérées et gérées par le client dans votre compte. Sélectionnez la politique à utiliser pour la politique d'autorisations ou choisissez Créer une politique pour ouvrir un nouvel onglet de navigateur et créer une nouvelle politique de bout en bout. Pour plus d'informations, consultez Création de politiques IAM. Une fois la politique créée, fermez cet onglet et revenez à l'onglet initial. Cochez la case en regard des politiques d'autorisations que vous souhaitez octroyer à toute personne endossant le rôle. Si vous préférez,

vous pouvez ne sélectionner aucune stratégie pour le moment, puis les attacher au rôle ultérieurement. Par défaut, un rôle ne dispose d'aucune autorisation.

9. (Facultatif) Définissez une limite d'autorisations. Il s'agit d'une fonctionnalité avancée.

   Ouvrez la section Set permissions boundary (Définir une limite d'autorisations) et choisissez Use a permissions boundary to control the maximum role permissions (Utiliser une limite d'autorisations pour contrôler le nombre maximum d'autorisations de rôle). Sélectionnez la politique à utiliser comme limite d'autorisations.

10. Choisissez Suivant.

11. Dans le champ Role name (Nom de rôle), saisissez un nom pour votre rôle. Les noms de rôles doivent être uniques au sein de votre Compte AWS. Lorsqu'un nom de rôle est utilisé dans une politique ou dans le cadre d'un ARN, il est sensible à la casse. Lorsque le nom d'un rôle apparaît aux clients dans la console, par exemple lors du processus de connexion, il n'est pas sensible à la casse. Différentes entités peuvent référencer le rôle et il n'est donc pas possible de modifier son nom après sa création.

12. (Facultatif) Pour Description, saisissez une description pour le nouveau rôle.

13. Choisissez Edit (Modifier) dans les sections Step 1: Select trusted entities (Étape 1 : sélection d'entités de confiance) ou Step 2: Add permissions (Étape 2 : ajouter des autorisations) pour modifier les cas d'utilisation et les autorisations pour le rôle. Vous serez renvoyé aux pages précédentes pour effectuer les modifications.

14. (Facultatif) Ajoutez des métadonnées au rôle en associant les identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Balisage des ressources IAM.

15. Passez en revue les informations du rôle, puis choisissez Créer un rôle.

> **Important**
>
> N'oubliez pas que ceci ne représente que la première moitié de la configuration requise. Vous devez également accorder aux utilisateurs individuels du compte approuvé des autorisations permettant de basculer vers le rôle dans la console ou d'endosser le rôle par programmation. Pour plus d'informations sur cette étape, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle.

Création d'un rôle IAM (AWS CLI)

La création d'un rôle à partir de AWS CLI implique plusieurs étapes. Lorsque vous utilisez la console pour créer un rôle, la plupart des étapes sont effectuées pour vous, mais AWS CLI vous devez effectuer chaque étape vous-même de manière explicite. Vous devez créer le rôle et lui attribuer une politique d'autorisations. Vous pouvez également définir la limite d'autorisations pour votre rôle.

Pour créer un rôle pour l'accès entre comptes (AWS CLI)

1. Créez un rôle : aws iam create-role
2. Associez une politique d'autorisations gérées au rôle : aws iam attach-role-policy

   or

   Créez une politique d'autorisation intégrée pour le rôle : aws iam put-role-policy
3. (Facultatif) Ajoutez des attributs personnalisés au rôle en associant des balises : aws iam tag-role

   Pour plus d'informations, consultez Gestion des balises sur les rôles IAM (AWS CLI ou AWS API).
4. (Facultatif) Définissez la limite des autorisations pour le rôle : aws iam put-role-permissions-boundary

   Une limite d'autorisations contrôle les autorisations maximum dont un rôle peut disposer. Les limites d'autorisations constituent une AWS fonctionnalité avancée.

L'exemple suivant illustre les deux premières étapes les plus courantes pour créer un rôle entre comptes dans un environnement simple. Cet exemple permet à tout utilisateur du compte `123456789012` d'endosser le rôle et d'afficher le compartiment Amazon S3 `example_bucket`. Cet exemple suppose également que vous utilisiez un ordinateur client exécutant Windows et que vous ayez déjà configuré votre interface de ligne de commande à l'aide des informations d'identification et de la région de votre compte. Pour plus d'informations, voir Configuration de l'interface de ligne de AWS commande.

Dans cet exemple, incluez la politique de confiance suivante dans la première commande lors de la création du rôle. Cette politique de confiance permet aux utilisateurs du compte `123456789012` d'endosser le rôle à l'aide de l'opération `AssumeRole`, mais uniquement s'ils fournissent l'authentification MFA à l'aide des paramètres `SerialNumber` et `TokenCode`. Pour plus

d'informations sur l'authentification MFA, consultez Utilisation de l'authentification multifactorielle (MFA) dans AWS.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
          "Action": "sts:AssumeRole",
          "Condition": { "Bool": { "aws:MultiFactorAuthPresent": "true" } }
      }
  ]
}
```

> **Important**
>
> Si votre élément `Principal` contient l'ARN d'un rôle ou utilisateur IAM spécifique, alors cet ARN devient un ID du principal unique lorsque la politique est enregistrée. Cela permet de réduire le risque d'escalade des autorisations par la suppression et la nouvelle création du rôle ou de l'utilisateur. Cet ID n'est pas fréquent dans la console, car il existe également une transformation inverse, pour revenir à l'ARN, lorsque la politique d'approbation est affichée. Toutefois, si vous supprimez le rôle ou l'utilisateur, l'ID principal apparaît dans la console car il n'est plus AWS possible de le mapper à un ARN. Par conséquent, si vous supprimez et recréez un utilisateur ou rôle référencé dans l'élément `Principal` d'une politique de confiance, vous devez modifier le rôle afin de remplacer l'ARN.

Lorsque vous utilisez la deuxième commande, vous devez attacher au rôle une politique gérée existante. La politique d'autorisations suivante permet à toute personne endossant le rôle d'exécuter uniquement l'action `ListBucket` sur le compartiment Amazon S3 `example_bucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Action": "s3:ListBucket",
          "Resource": "arn:aws:s3:::example_bucket"
      }
```

```
  ]
}
```

Pour créer ce rôle `Test-UserAccess-Role`, vous devez d'abord enregistrer la précédente politique de confiance avec le nom `trustpolicyforacct123456789012.json` dans le dossier `policies` de votre disque local `C:`. Enregistrez ensuite la politique d'autorisation précédente en tant que politique gérée par le client dans votre Compte AWS nom`PolicyForRole`. Vous pouvez ensuite utiliser les commandes suivantes pour créer le rôle et attacher la politique gérée.

```
# Create the role and attach the trust policy file that allows users in the specified
 account to assume the role.
$ aws iam create-role --role-name Test-UserAccess-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforacct123456789012.json

# Attach the permissions policy (in this example a managed policy) to the role to
 specify what it is allowed to do.
$ aws iam attach-role-policy --role-name Test-UserAccess-Role --policy-arn
 arn:aws:iam::123456789012:policy/PolicyForRole
```

> **Important**
>
> N'oubliez pas que ceci ne représente que la première moitié de la configuration requise. Vous devez également accorder à des utilisateurs individuels du compte approuvé les autorisations permettant de changer de rôle. Pour plus d'informations sur cette étape, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle.

Une fois que vous avez créé le rôle et que vous lui avez accordé les autorisations nécessaires pour effectuer des AWS tâches ou accéder aux AWS ressources, tous les utilisateurs du `123456789012` compte peuvent assumer le rôle. Pour plus d'informations, consultez Assumer un rôle IAM (AWS CLI).

Création d'un rôle IAM (AWS API)

La création d'un rôle à partir de l' AWS API implique plusieurs étapes. Lorsque vous utilisez la console pour créer un rôle, la plupart des étapes sont exécutées automatiquement pour vous, mais avec l'API vous devez exécuter explicitement chaque étape vous-même. Vous devez créer le rôle et lui attribuer une politique d'autorisations. Vous pouvez également définir la limite d'autorisations pour votre rôle.

Pour créer un rôle dans le code (AWS API)

1. Créez un rôle : CreateRole

   Vous pouvez spécifier un emplacement de fichier pour la politique d'approbation du rôle.
2. Associez une politique d'autorisation gérée au rôle : AttachRolePolicy

   or

   Créez une politique d'autorisation intégrée pour le rôle : PutRolePolicy

   > **Important**
   >
   > N'oubliez pas que ceci ne représente que la première moitié de la configuration requise. Vous devez également accorder à des utilisateurs individuels du compte approuvé les autorisations permettant de changer de rôle. Pour plus d'informations sur cette étape, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle.
3. (Facultatif) Ajoutez des attributs personnalisés à l'utilisateur en attachant des balises : TagRole

   Pour plus d'informations, consultez Gestion des balises sur les utilisateurs IAM (AWS CLI ou AWS API).
4. (Facultatif) Définissez la limite des autorisations pour le rôle : PutRolePermissionsBoundary

   Une limite d'autorisations contrôle les autorisations maximum dont un rôle peut disposer. Les limites d'autorisations constituent une AWS fonctionnalité avancée.

Une fois que vous avez créé le rôle et que vous lui avez accordé les autorisations nécessaires pour effectuer des AWS tâches ou accéder aux AWS ressources, vous devez accorder des autorisations aux utilisateurs du compte pour leur permettre d'assumer le rôle. Pour plus d'informations sur l'endossement d'un rôle, consultez Passage à un rôle IAM (AWS API).

Création d'un rôle IAM (AWS CloudFormation)

Pour plus d'informations sur la création d'un rôle IAM dans AWS CloudFormation, consultez la référence sur les ressources et les propriétés et les exemples dans le guide de l'AWS CloudFormation utilisateur.

Pour plus d'informations sur les modèles IAM dans AWS CloudFormation, consultez les extraits de AWS Identity and Access Management modèles dans le Guide de l'AWS CloudFormation utilisateur.

Création d'un rôle pour la délégation d'autorisations à un service AWS

De nombreux AWS services nécessitent que vous utilisiez des rôles pour permettre au service d'accéder aux ressources d'autres services en votre nom. Un rôle qu'un service endosse pour effectuer des actions en votre nom s'appelle un rôle de service. Lorsqu'un rôle remplit un objectif spécial pour un service, il est défini comme un rôle de service pour les instances EC2 (par exemple) ou un rôle lié à un service. Pour savoir quels services prennent en charge par les rôles de service liés à un service ou déterminer si un service prend en charge une forme d'informations d'identification temporaires, consultez AWS services qui fonctionnent avec IAM. Pour découvrir comment un service individuel utilise des rôles, choisissez son nom dans le tableau pour afficher la documentation de ce service.

Lorsque vous définissez l'`PassRole`autorisation, vous devez vous assurer qu'un utilisateur ne transfère pas un rôle pour lequel le rôle dispose de plus d'autorisations que vous ne le souhaitez. Par exemple, Alice n'est peut-être pas autorisée à effectuer des actions Amazon S3. Si Alice pouvait transmettre un rôle à un service qui autorise les actions Amazon S3, le service pourrait effectuer des actions Amazon S3 pour le compte d'Alice lors de l'exécution de la tâche.

Pour plus d'informations sur la façon dont les rôles aident à déléguer des autorisations, consultez Termes et concepts relatifs aux rôles.

Autorisations de fonction de service

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur ou un rôle) de créer ou modifier un rôle lié à un service.

> **Note**
>
> L'ARN d'un rôle lié à un service comprend un principal de service indiqué dans les politiques suivantes comme *`SERVICE-NAME`*`.amazonaws.com`. N'essayez pas de deviner le principal de service, car il est sensible à la casse et le format peut varier entre les services AWS . Pour afficher le principal de service d'un service, consultez la documentation de son rôle lié à un service.

Pour permettre à une entité IAM de créer un rôle spécifique lié à un service

Ajoutez la politique suivante à l'entité IAM qui doit créer le rôle lié à un service. Cette politique vous permet de créer un rôle lié au service spécifié et avec un nom spécifique. Vous pouvez ensuite attacher des politiques en ligne ou gérées à ce rôle.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:CreateRole",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        }
    ]
}
```

Pour permettre à une entité IAM de créer n"importe quel rôle lié à un service

AWS recommande de n'autoriser que les utilisateurs administratifs à créer un rôle de service. Une personne disposant des autorisations nécessaires pour créer un rôle et attacher n'importe quelle politique peut augmenter ses propres autorisations. Au lieu de cela, créez une politique qui l'autorise à créer uniquement les rôles dont elle a besoin ou de demander à un administrateur de créer la fonction du service en son nom.

Pour associer une politique permettant à un administrateur d'accéder à votre intégralité Compte AWS, utilisez la politique AdministratorAccess AWS gérée.

Pour permettre à une entité IAM de modifier un rôle lié à un service

Ajoutez la politique suivante à l'entité IAM qui doit modifier le rôle lié à un service.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EditSpecificServiceRole",
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:DeleteRolePolicy",
                "iam:DetachRolePolicy",
                "iam:GetRole",
                "iam:GetRolePolicy",
```

```
                "iam:ListAttachedRolePolicies",
                "iam:ListRolePolicies",
                "iam:PutRolePolicy",
                "iam:UpdateRole",
                "iam:UpdateRoleDescription"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        },
        {
            "Sid": "ViewRolesAndPolicies",
            "Effect": "Allow",
            "Action": [
                "iam:GetPolicy",
                "iam:ListRoles"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour permettre à une entité IAM de supprimer un rôle spécifique lié à un service

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit supprimer le rôle lié à un service spécifié.

```
{
    "Effect": "Allow",
    "Action": "iam:DeleteRole",
    "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
}
```

Pour permettre à une entité IAM de supprimer un rôle de service

AWS recommande de n'autoriser que les utilisateurs administratifs à supprimer un rôle de service. Au lieu de cela, créez une politique qui leur permet de supprimer uniquement les rôles dont ils ont besoin ou de demander à un administrateur de supprimer le rôle de service en leur nom.

Pour associer une politique permettant à un administrateur d'accéder à votre intégralité Compte AWS, utilisez la politique AdministratorAccess AWS gérée.

Création d'un rôle pour un AWS service (console)

Vous pouvez utiliser le AWS Management Console pour créer un rôle pour un service. Puisque certains services prennent en charge plus d'un rôle de service, consultez la documentation AWS de votre service pour savoir quel cas d'utilisation choisir. Vous pouvez apprendre à attribuer les politiques d'approbation et d'autorisation nécessaires au rôle, afin que le service puisse endosser le rôle à votre place. Les étapes que vous pouvez utiliser pour contrôler les autorisations de votre rôle peuvent varier, selon la façon dont le service définit les cas d'utilisation et si vous créez un rôle lié à un service.

Pour créer un rôle pour une Service AWS (console IAM)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.
4. Pour Service ou cas d'utilisation, choisissez un service, puis choisissez le cas d'utilisation. Les cas d'utilisation sont définis par le service pour inclure la politique d'approbation nécessaire au service.
5. Choisissez Suivant.
6. Pour les politiques d'autorisations, les options dépendent du cas d'utilisation que vous avez sélectionné :

   - Si le service définit les autorisations pour le rôle, vous ne pouvez pas sélectionner de politiques d'autorisation.
   - Choisissez parmi un ensemble limité de politiques d'autorisation.
   - Choisissez parmi toutes les politiques d'autorisation.
   - Sélectionnez aucune politique d'autorisation, créez les politiques une fois le rôle créé, puis attachez les politiques au rôle.
7. (Facultatif) Définissez une limite d'autorisations. Il s’agit d’une fonctionnalité avancée disponible pour les fonctions de service, mais pas pour les rôles liés à un service.

   a. Ouvrez la section Définir les limites des autorisations, puis choisissez Utiliser une limite d'autorisations pour contrôler le nombre maximal d'autorisations de rôle.

      IAM inclut une liste des politiques AWS gérées et gérées par le client dans votre compte.

b. Sélectionnez la politique à utiliser comme limite d'autorisations.

8. Choisissez Suivant.
9. Pour le nom du rôle, les options dépendent du service :

   - Si le service définit le nom du rôle, vous ne pouvez pas le modifier.
   - Si le service définit un préfixe pour le nom du rôle, vous pouvez saisir un suffixe facultatif.
   - Si le service ne définit pas le nom du rôle, vous pouvez le nommer.

   > **Important**
   >
   > Lorsque vous nommez un rôle, tenez compte des points suivants :
   >
   > - Les noms de rôles doivent être uniques au sein du Compte AWS vôtre et ne peuvent pas être rendus uniques au cas par cas.
   >
   >   Par exemple, ne créez pas de rôles nommés à la fois **`PRODROLE`** et**`prodrole`**. Lorsqu'un nom de rôle est utilisé dans une politique ou dans le cadre d'un ARN, le nom du rôle distingue les majuscules et minuscules, mais lorsqu'un nom de rôle apparaît aux clients dans la console, par exemple pendant le processus de connexion, le nom du rôle ne distingue pas les majuscules et minuscules.
   > - Vous ne pouvez pas modifier le nom du rôle une fois qu'il a été créé car d'autres entités peuvent y faire référence.

10. (Facultatif) Dans Description, entrez une description pour le rôle.
11. (Facultatif) Pour modifier les cas d'utilisation et les autorisations du rôle, dans les sections Étape 1 : Sélection des entités de confiance ou Étape 2 : Ajouter des autorisations, choisissez Modifier.
12. (Facultatif) Pour identifier, organiser ou rechercher le rôle, ajoutez des balises sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la rubrique Balisage des ressources IAM dans le Guide de l'utilisateur IAM.
13. Passez en revue les informations du rôle, puis choisissez Create role (Créer un rôle).

Création d'un rôle pour un service (AWS CLI)

La création d'un rôle à partir de AWS CLI implique plusieurs étapes. Lorsque vous utilisez la console pour créer un rôle, la plupart des étapes sont effectuées pour vous, mais AWS CLI vous devez effectuer chaque étape vous-même de manière explicite. Vous devez créer le rôle et lui attribuer une politique d'autorisations. Si le service concerné est Amazon EC2 vous devez également créer

un profil d'instance et lui ajouter le rôle. Vous pouvez également définir la limite d'autorisations pour votre rôle.

Pour créer un rôle pour un AWS service à partir du AWS CLI

1. La commande `create-role` suivante crée un rôle nommé Rôle test et lui attache une politique de confiance :

   ```
   aws iam create-role --role-name Test-Role --assume-role-policy-document
   file://Test-Role-Trust-Policy.json
   ```

2. Associez une politique d'autorisations gérées au rôle : aws iam attach-role-policy.

   Par exemple, la commande `attach-role-policy` suivante attache la politique gérée par AWS , nommée `ReadOnlyAccess`, au rôle IAM nommé `ReadOnlyRole` :

   ```
   aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
   ReadOnlyAccess --role-name ReadOnlyRole
   ```

   or

   Créez une politique d'autorisation intégrée pour le rôle : aws iam put-role-policy

   Pour ajouter une politique d'autorisations en ligne, reportez-vous à l'exemple suivant :

   ```
   aws iam put-role-policy --role-name Test-Role --policy-name
   ExamplePolicy --policy-document file://AdminPolicy.json
   ```

3. (Facultatif) Ajoutez des attributs personnalisés au rôle en associant des balises : aws iam tag-role

   Pour plus d'informations, consultez Gestion des balises sur les rôles IAM (AWS CLI ou AWS API).

4. (Facultatif) Définissez la limite des autorisations pour le rôle : aws iam put-role-permissions-boundary

   Une limite d'autorisations contrôle les autorisations maximum dont un rôle peut disposer. Les limites d'autorisations constituent une AWS fonctionnalité avancée.

Si vous comptez utiliser le rôle avec Amazon EC2 ou un autre AWS service utilisant Amazon EC2, vous devez le stocker dans un profil d'instance. Un profil d'instance est un conteneur pour un rôle que

vous pouvez attacher à une instance Amazon EC2 lorsqu'elle est lancée. Un profil d'instance peut contenir un rôle uniquement et cette limite ne peut pas être augmentée. Si vous créez le rôle à l'aide du AWS Management Console, le profil d'instance est créé pour vous sous le même nom que le rôle. Pour plus d'informations sur les profils d'instance, consultez Utilisation de profils d'instance. Pour plus d'informations sur le lancement d'une instance EC2 avec un rôle, consultez la section Contrôle de l'accès aux ressources Amazon EC2 dans le guide de l'utilisateur Amazon EC2.

Pour créer un profil d'instance et y stocker le rôle (AWS CLI)

1. Création d'un profil d'instance : aws iam create-instance-profile
2. Ajoutez le rôle au profil d'instance : aws iam add-role-to-instance -profile

L' AWS CLI exemple de commande ci-dessous illustre les deux premières étapes de création d'un rôle et d'attribution d'autorisations. Il présente également les deux étapes de la création d'un profil d'instance et de l'ajout du rôle au profil. Cet exemple de politique de confiance permet au service Amazon EC2 d'endosser le rôle et d'afficher le compartiment Amazon S3 `example_bucket`. L'exemple suppose également que vous utilisiez un ordinateur client exécutant Windows et que vous ayez déjà configuré votre interface de ligne de commande avec vos informations d'identification et votre région. Pour plus d'informations, voir Configuration de l'interface de ligne de AWS commande.

Dans cet exemple, incluez la politique de confiance suivante dans la première commande lors de la création du rôle. Cette politique de confiance permet au service Amazon EC2 d'endosser le rôle.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"Service": "ec2.amazonaws.com"},
    "Action": "sts:AssumeRole"
  }
}
```

Lorsque vous utilisez la deuxième commande, vous devez attacher une politique d'autorisations au rôle. L'exemple de politique d'autorisations suivant permet au rôle d'exécuter uniquement l'action `ListBucket` sur le compartiment Amazon S3 `example_bucket`.

```
{
  "Version": "2012-10-17",
  "Statement": {
```

```
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Pour créer ce rôle `Test-Role-for-EC2`, vous devez d'abord enregistrer la politique de confiance précédente avec le nom `trustpolicyforec2.json` et la politique d'autorisations précédente avec le nom `permissionspolicyforec2.json` dans le répertoire `policies` de votre disque local `C:`. Vous pouvez ensuite utiliser les commandes suivantes pour créer le rôle, attacher la politique, créer le profil d'instance et ajouter le rôle au profil d'instance.

```
# Create the role and attach the trust policy that allows EC2 to assume this role.
$ aws iam create-role --role-name Test-Role-for-EC2 --assume-role-policy-document
 file://C:\policies\trustpolicyforec2.json

# Embed the permissions policy (in this example an inline policy) to the role to
 specify what it is allowed to do.
$ aws iam put-role-policy --role-name Test-Role-for-EC2 --policy-name Permissions-
Policy-For-Ec2 --policy-document file://C:\policies\permissionspolicyforec2.json

# Create the instance profile required by EC2 to contain the role
$ aws iam create-instance-profile --instance-profile-name EC2-ListBucket-S3

# Finally, add the role to the instance profile
$ aws iam add-role-to-instance-profile --instance-profile-name EC2-ListBucket-S3 --
role-name Test-Role-for-EC2
```

Lorsque vous lancez l'instance EC2, spécifiez le nom du profil de l'instance sur la page Configurer les détails de l'instance si vous utilisez la AWS console. Si vous utilisez la commande CLI `aws ec2 run-instances`, précisez le paramètre `--iam-instance-profile`.

Création d'un rôle pour un service (API AWS)

La création d'un rôle à partir de l' AWS API implique plusieurs étapes. Lorsque vous utilisez la console pour créer un rôle, la plupart des étapes sont exécutées automatiquement pour vous, mais avec l'API vous devez exécuter explicitement chaque étape vous-même. Vous devez créer le rôle et lui attribuer une politique d'autorisations. Si le service concerné est Amazon EC2 vous devez également créer un profil d'instance et lui ajouter le rôle. Vous pouvez également définir la limite d'autorisations pour votre rôle.

Pour créer un rôle pour un AWS service (AWS API)

1. Créez un rôle : CreateRole

   Vous pouvez spécifier un emplacement de fichier pour la politique d'approbation du rôle.
2. Associer une politique d'autorisations gérées au rôle : AttachRolePolitique

   or

   Créez une politique d'autorisation intégrée pour le rôle : PutRole Politique
3. (Facultatif) Ajoutez des attributs personnalisés à l'utilisateur en attachant des balises : TagRole

   Pour plus d'informations, consultez Gestion des balises sur les utilisateurs IAM (AWS CLI ou AWS API).
4. (Facultatif) Définissez la limite des autorisations pour le rôle : PutRolePermissionsBoundary

   Une limite d'autorisations contrôle les autorisations maximum dont un rôle peut disposer. Les limites d'autorisations constituent une AWS fonctionnalité avancée.

Si vous comptez utiliser le rôle avec Amazon EC2 ou un autre AWS service utilisant Amazon EC2, vous devez le stocker dans un profil d'instance. Un profil d'instance est un conteneur pour un rôle. Chaque profil d'instance peut contenir un rôle uniquement et cette limite ne peut pas être augmentée. Si vous créez le rôle dans le AWS Management Console, le profil d'instance est créé pour vous sous le même nom que le rôle. Pour plus d'informations sur les profils d'instance, consultez Utilisation de profils d'instance. Pour plus d'informations sur le lancement d'une instance Amazon EC2 avec un rôle, consultez la section Contrôle de l'accès aux ressources Amazon EC2 dans le guide de l'utilisateur Amazon EC2.

Pour créer un profil d'instance et y stocker le rôle (AWS API)

1. Création d'un profil d'instance : CreateInstanceProfil
2. Ajoutez le rôle au profil de l'instance : AddRoleToInstanceProfil

Création d'un rôle pour un fournisseur d'identité tiers (fédération)

Vous pouvez utiliser des fournisseurs d'identité au lieu de créer des utilisateurs IAM dans votre Compte AWS. Avec un fournisseur d'identité (IdP), vous pouvez gérer vos identités d'utilisateurs en dehors de l'extérieur AWS et leur donner les autorisations d'accéder aux AWS ressources de

votre compte. Pour plus d'informations sur la fédération et les fournisseurs d'identité, consultez Fournisseurs d'identité et fédération.

Création d'un rôle pour les utilisateurs fédérés (console)

Les procédures permettant de créer un rôle pour les utilisateurs fédérés dépendent des fournisseurs tiers choisis :

- Pour OpenID Connect (OIDC), voir. Création d'un rôle pour la fédération OpenID Connect (console)
- Pour SAML 2.0, consultez Création d'un rôle pour la fédération SAML 2.0 (console).

Création d'un rôle pour l'accès fédéré (AWS CLI)

Les étapes à suivre pour la création d'un rôle pour les fournisseurs d'identité pris en charge (OIDC ou SAML) à partir de la AWS CLI sont identiques. La différence porte sur le contenu de la politique d'approbation que vous créez dans les étapes préalables requises. Commencez par suivre les étapes de la section Prérequis pour le type de fournisseur que vous utilisez :

- Pour un fournisseur OIDC, consultez Conditions préalables à la création d'un rôle pour OIDC.
- Pour un fournisseur SAML, consultez Prérequis pour la création d'un rôle pour SAML.

La création d'un rôle à partir de AWS CLI implique plusieurs étapes. Lorsque vous utilisez la console pour créer un rôle, la plupart des étapes sont effectuées pour vous, mais AWS CLI vous devez effectuer chaque étape vous-même de manière explicite. Vous devez créer le rôle et lui attribuer une politique d'autorisations. Vous pouvez également définir la limite d'autorisations pour votre rôle.

Pour créer un rôle pour la fédération d'identité (AWS CLI)

1. Créez un rôle : aws iam create-role
2. Associez une politique d'autorisation au rôle : aws iam attach-role-policy

   or

   Créez une politique d'autorisation intégrée pour le rôle : aws iam put-role-policy
3. (Facultatif) Ajoutez des attributs personnalisés au rôle en associant des balises : aws iam tag-role

Pour plus d'informations, consultez Gestion des balises sur les rôles IAM (AWS CLI ou AWS API).

4. (Facultatif) Définissez la limite des autorisations pour le rôle : aws iam put-role-permissions-boundary

   Une limite d'autorisations contrôle les autorisations maximum dont un rôle peut disposer. Les limites d'autorisations constituent une AWS fonctionnalité avancée.

L'exemple suivant illustre les deux premières étapes les plus courantes pour créer un rôle de fournisseur d'identité dans un environnement simple. Cet exemple permet à tout utilisateur du compte `123456789012` d'endosser le rôle et d'afficher le compartiment Amazon S3 `example_bucket`. Cet exemple suppose également que vous exécutez le AWS CLI sur un ordinateur exécutant Windows et que vous l'avez déjà configuré AWS CLI avec vos informations d'identification. Pour plus d'informations, consultez Configuration de l' AWS Command Line Interface.

L'exemple de politique de confiance suivant est conçu pour une application mobile si l'utilisateur se connecte à l'aide d'Amazon Cognito. Dans cet exemple, *`us-east:12345678- ffff-ffff-ffff-123456`* représente l'ID du pool d'identités attribué par Amazon Cognito.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "RoleForCognito",
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east:12345678-ffff-ffff-ffff-123456"}}
    }
}
```

La politique d'autorisations suivante permet à toute personne endossant le rôle d'exécuter uniquement l'action `ListBucket` sur le compartiment Amazon S3 `example_bucket`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
```

```
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Pour créer ce rôle `Test-Cognito-Role`, vous devez d'abord enregistrer la politique de confiance précédente avec le nom `trustpolicyforcognitofederation.json` et la politique d'autorisations précédente avec le nom `permspolicyforcognitofederation.json` dans le dossier `policies` de votre disque local `C:`. Vous pouvez ensuite utiliser les commandes suivantes pour créer le rôle et attacher la politique en ligne.

```
# Create the role and attach the trust policy that enables users in an account to
 assume the role.
$ aws iam create-role --role-name Test-Cognito-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforcognitofederation.json

# Attach the permissions policy to the role to specify what it is allowed to do.
aws iam put-role-policy --role-name Test-Cognito-Role --policy-name
 Perms-Policy-For-CognitoFederation --policy-document file://C:\policies
\permspolicyforcognitofederation.json
```

Création d'un rôle pour l'accès fédéré (AWS API)

Les étapes à suivre pour la création d'un rôle pour les fournisseurs d'identité pris en charge (OIDC ou SAML) à partir de la AWS CLI sont identiques. La différence porte sur le contenu de la politique d'approbation que vous créez dans les étapes préalables requises. Commencez par suivre les étapes de la section Prérequis pour le type de fournisseur que vous utilisez :

- Pour un fournisseur OIDC, consultez Conditions préalables à la création d'un rôle pour OIDC.
- Pour un fournisseur SAML, consultez Prérequis pour la création d'un rôle pour SAML.

Pour créer un rôle pour la fédération d'identités (AWS API)

1. Créez un rôle : CreateRole
2. Associez une politique d'autorisation au rôle : AttachRolePolicy

   or

   Créez une politique d'autorisation intégrée pour le rôle : PutRolePolicy
3. (Facultatif) Ajoutez des attributs personnalisés à l'utilisateur en attachant des balises : TagRole

Pour plus d'informations, consultez Gestion des balises sur les utilisateurs IAM (AWS CLI ou AWS API).

4. (Facultatif) Définissez la limite des autorisations pour le rôle : PutRolePermissionsBoundary

   Une limite d'autorisations contrôle les autorisations maximum dont un rôle peut disposer. Les limites d'autorisations constituent une AWS fonctionnalité avancée.

Création d'un rôle pour la fédération OpenID Connect (console)

Vous pouvez utiliser les fournisseurs d'identité fédérés OpenID Connect (OIDC) au lieu de créer AWS Identity and Access Management des utilisateurs dans votre. Compte AWS Avec un fournisseur d'identité (IdP), vous pouvez gérer vos identités d'utilisateurs en dehors de l'extérieur AWS et leur donner les autorisations d'accéder aux AWS ressources de votre compte. Pour plus d'informations sur la fédération et IdPs, voirFournisseurs d'identité et fédération.

Conditions préalables à la création d'un rôle pour OIDC

Avant de créer un rôle pour la fédération OIDC, vous devez d'abord suivre les étapes préalables suivantes.

Pour préparer la création d'un rôle pour la fédération OIDC

1. Inscrivez-vous auprès d'un ou plusieurs services offrant une identité OIDC fédérée. Si vous créez une application qui a besoin d'accéder à vos AWS ressources, vous devez également configurer votre application avec les informations du fournisseur. Ainsi, le fournisseur vous communique un ID d'application ou de public unique pour votre application. (Les fournisseurs utilisent une terminologie différente pour ce processus. Ce guide utilise le terme configurer pour désigner le processus d'identification de votre application auprès du fournisseur.) Vous pouvez configurer plusieurs applications auprès de chaque fournisseur, ou plusieurs fournisseurs avec une seule application. Consultez les informations sur l'utilisation des fournisseurs d'identité comme suit :

   - Login with Amazon Developer Center
   - Add Facebook Login to Your App or Website sur le site des développeurs Facebook.
   - Using OAuth 2.0 for Login (OpenID Connect) sur le site des développeurs Google.
2. Après avoir reçu les informations requises de la part de l'IdP, créez un IdP dans IAM. Pour plus d'informations, consultez Création d'un fournisseur d'identité OpenID Connect (OIDC) dans IAM.

> **Important**
>
> Si vous utilisez un fournisseur d'identités OIDC de Google, Facebook ou Amazon Cognito, ne créez pas d'IdP IAM distinct dans l' AWS Management Console. Ces fournisseurs d'identité OIDC sont déjà intégrés AWS et sont à votre disposition. Ignorez cette étape et créez de nouveaux rôles à l'aide de votre IdP à l'étape suivante.

3. Préparez les politiques pour le rôle que les utilisateurs authentifiés auprès de l'IdP vont endosser. Comme n'importe quel rôle, celui d'une application mobile inclut deux politiques. L'une est la politique de confiance qui spécifie la personne capable d'endosser le rôle. L'autre est la politique d'autorisation qui spécifie les actions et les ressources AWS auxquelles l'application mobile peut accéder ou non.

   Pour le Web IdPs, nous vous recommandons d'utiliser Amazon Cognito pour gérer les identités. Dans ce cas, utilisez une politique de confiance semblable à cet exemple.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {
            "StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east-2:12345678-abcd-abcd-abcd-123456"},
            "ForAnyValue:StringLike": {"cognito-identity.amazonaws.com:amr":
 "unauthenticated"}
        }
    }
}
```

   Remplacez `us-east-2:12345678-abcd-abcd-abcd-123456` par l'ID du groupe d'identités qu'Amazon Cognito vous assigne.

   Si vous configurez manuellement un IdP OIDC, lorsque vous créez la politique de confiance, vous devez utiliser trois valeurs garantissant que seule votre application peut assumer le rôle :

   - Pour l'élément `Action`, utilisez l'action `sts:AssumeRoleWithWebIdentity`.

- Pour l'élément `Principal`, utilisez la chaîne `{"Federated":`*`providerUrl/providerArn`*`}`.
  - Pour certains OIDC courants IdPs, il *`providerUrl`* s'agit d'une URL. Les exemples suivants incluent des méthodes permettant de spécifier le principal pour certaines méthodes courantes IdPs :

    ```
    "Principal":{"Federated":"cognito-identity.amazonaws.com"}
    ```

    ```
    "Principal":{"Federated":"www.amazon.com"}
    ```

    ```
    "Principal":{"Federated":"graph.facebook.com"}
    ```

    ```
    "Principal":{"Federated":"accounts.google.com"}
    ```

  - Pour les autres fournisseurs OIDC, utilisez l'Amazon Resource Name (ARN) du fournisseur d'identité OIDC créé dans Step 2, comme dans l'exemple suivant :

    ```
    "Principal":{"Federated":"arn:aws:iam::123456789012:oidc-provider/
    server.example.com"}
    ```

- Pour l'élément `Condition`, utilisez une condition `StringEquals` pour limiter les autorisations. Testez l'ID du groupe d'identités pour Amazon Cognito ou l'ID d'application pour les autres fournisseurs. L'ID du groupe d'identité doit correspondre à l'ID d'application obtenu lors de la configuration de l'application avec le fournisseur d'identité. Cette correspondance entre les ID permet de vérifier que la demande provient bien de votre application.

> **Note**
>
> Les rôles IAM pour les pools d'identités Amazon Cognito font confiance au `cognito-identity.amazonaws.com` principal du service pour assumer ce rôle. Les rôles de ce type doivent contenir au moins une clé de condition afin de limiter le nombre de personnes principales pouvant assumer ce rôle.
> Des considérations supplémentaires s'appliquent aux pools d'identités Amazon Cognito qui assument des rôles IAM entre comptes. Les politiques de confiance associées à ces rôles doivent accepter le principe du `cognito-identity.amazonaws.com` service et contenir la clé de `aud` condition permettant de limiter l'attribution de rôles aux utilisateurs issus des pools d'identités que vous souhaitez. Une politique qui fait confiance aux groupes d'identités Amazon Cognito sans cette condition crée le risque qu'un utilisateur d'un pool d'identités involontaire assume ce rôle. Pour plus d'informations, consultez la section Politiques de confiance

pour les rôles IAM dans le cadre de l'authentification de base (classique) dans le manuel Amazon Cognito Developer Guide.

Créez un élément de condition semblable à un des exemples suivants, en fonction du fournisseur d'identité que vous utilisez :

```
"Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud":
"us-east:12345678-ffff-ffff-ffff-123456"}}
```

```
"Condition": {"StringEquals": {"www.amazon.com:app_id":
"amzn1.application-oa2-123456"}}
```

```
"Condition": {"StringEquals": {"graph.facebook.com:app_id":
"111222333444555"}}
```

```
"Condition": {"StringEquals": {"accounts.google.com:aud":
"66677788899900pro0"}}
```

Pour les fournisseurs OIDC, utilisez l'URL complète du fournisseur d'identité OIDC avec la clé de contexte `aud`, comme dans exemple suivant :

```
"Condition": {"StringEquals": {"server.example.com:aud":
"appid_from_oidc_idp"}}
```

> **Note**
>
> Les valeurs du principal dans la politique de confiance pour le rôle sont propres à un fournisseur d'identité. Un rôle pour OIDC ne peut spécifier qu'un seul principal. Par conséquent, si l'application mobile autorise des utilisateurs à se connecter à partir de plusieurs fournisseurs d'identité, créez un rôle distinct pour chaque fournisseur d'identité que vous souhaitez prendre en charge. Créez des politiques de confiance distinctes pour chaque fournisseur d'identité.

Si un utilisateur utilise une application mobile pour se connecter à partir de Login with Amazon (Connexion avec Amazon, l'exemple suivant de politique de confiance s'appliquerait. Dans l'exemple, *amzn1.application-oa2-123456* représente l'ID d'application que Amazon

assigne lorsque vous avez configuré l'application à l'aide de Login with Amazon (Connexion avec Amazon).

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForLoginWithAmazon",
          "Effect": "Allow",
          "Principal": {"Federated": "www.amazon.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"www.amazon.com:app_id":
 "amzn1.application-oa2-123456"}}
      }]
  }
```

Si un utilisateur utilise une application mobile pour se connecter à partir de Facebook, l'exemple suivant de politique de confiance s'appliquerait. Dans cet exemple, *111222333444555* représente l'ID d'application que Facebook assigne.

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForFacebook",
          "Effect": "Allow",
          "Principal": {"Federated": "graph.facebook.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"graph.facebook.com:app_id":
 "111222333444555"}}
      }]
  }
```

Si un utilisateur utilise une application mobile pour se connecter à partir de Google, l'exemple suivant de politique de confiance s'appliquerait. Dans cet exemple, *666777888999000* représente l'ID d'application que Google assigne.

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForGoogle",
```

```
          "Effect": "Allow",
          "Principal": {"Federated": "accounts.google.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"accounts.google.com:aud":
 "666777888999000"}}
      }]
  }
```

Si un utilisateur utilise une application mobile pour se connecter depuis Amazon Cognito, l'exemple de politique de confiance suivant s'applique. Dans cet exemple, *us-east:12345678- ffff-ffff-ffff -123456* représente l'ID du pool d'identités attribué par Amazon Cognito.

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForCognito",
          "Effect": "Allow",
          "Principal": {"Federated": "cognito-identity.amazonaws.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east:12345678-ffff-ffff-ffff-123456"}}
      }]
  }
```

Création d'un rôle pour l'OIDC

Après avoir exécuté les prérequis, vous pouvez créer le rôle dans IAM. La procédure suivante décrit comment créer le rôle pour la fédération OIDC dans le AWS Management Console. Pour créer un rôle à partir de l' AWS API AWS CLI or, consultez les procédures surCréation d'un rôle pour un fournisseur d'identité tiers (fédération).

> **Important**
>
> Si vous utilisez Amazon Cognito, utilisez la console Amazon Cognito pour configurer les rôles. Sinon, utilisez la console IAM pour créer un rôle pour la fédération OIDC.

Pour créer un rôle IAM pour la fédération OIDC

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Roles (Rôles), puis Create role (Créer un rôle).
3. Choisissez le type de rôle OIDC.
4. Pour Identity provider (Fournisseur d'identité), choisissez l'IdP de votre rôle :

   - Si vous créez un rôle pour un fournisseur d'identité Web individuel, choisissez entre Login with Amazon (Connexion avec Amazon), Facebook ou Google.

     > **Note**
     >
     > Vous devez créer un rôle distinct pour chaque IdP que vous souhaitez prendre en charge.

   - Si vous souhaitez créer un rôle de scénario avancé pour Amazon Cognito, choisissez Amazon Cognito.

     > **Note**
     >
     > Vous ne devez créer manuellement un rôle à utiliser avec Amazon Cognito que lorsque vous travaillez sur un scénario avancé. Sinon, Amazon Cognito peut créer des rôles pour vous. Pour plus d'informations sur Amazon Cognito, consultez Fournisseurs d'identité externes - Pools d’identités (identités fédérées) dans le Manuel du développeur Amazon Cognito.

   - Si vous souhaitez créer un rôle pour GitHub Actions, vous devez commencer par ajouter le fournisseur GitHub OIDC à IAM. Après avoir ajouté le fournisseur GitHub OIDC à IAM, choisissez token.actions.githubusercontent.com.

     > **Note**
     >
     > Pour plus d'informations sur la façon de AWS configurer GitHub l'OIDC de confiance en tant qu'identité fédérée, consultez GitHub Docs - Configuring OpenID Connect in Amazon Web Services. Pour plus d'informations sur les meilleures pratiques en matière de limitation de l'accès aux rôles associés à l'IdP IAM GitHub pour,

Configuration d'un rôle pour le fournisseur d'identité GitHub OIDC consultez cette page.

5. Saisissez l'identifiant de votre application. L'étiquette de l'identifiant varie selon le fournisseur que vous choisissez :

   - Si vous souhaitez créer un rôle pour Login with Amazon (Connexion avec Amazon), saisissez l'ID d'application dans le champ Application ID (ID d'application).
   - Si vous souhaitez créer un rôle pour Facebook, saisissez l'ID d'application dans le champ Application ID (ID d'application).
   - Si vous souhaitez créer un rôle pour Google, saisissez le nom du public cible dans le champ Audience (Public cible).
   - Si vous souhaitez créer un rôle pour Amazon Cognito, saisissez l'ID du groupe d'identités que vous avez créé pour vos applications Amazon Cognito dans le champ Identity Pool ID (ID du pool d'identités).
   - Si vous souhaitez créer un rôle pour GitHub Actions, entrez les informations suivantes :
     - Pour Audience, choisissez `sts.amazonaws.com`.
     - Pour GitHub organisation, entrez le nom de votre GitHub organisation. Le nom de GitHub l'organisation est obligatoire et doit être alphanumérique, y compris les tirets (-). Vous ne pouvez pas utiliser de caractères génériques (* et ?) dans le nom GitHub de l'organisation.
     - (Facultatif) Pour le GitHub référentiel, entrez le nom du GitHub référentiel. Si vous ne précisez pas de valeur, la valeur par défaut devient un caractère de remplacement (*).
     - (Facultatif) Pour GitHub la branche, entrez le nom de la GitHub branche. Si vous ne précisez pas de valeur, la valeur par défaut devient un caractère de remplacement (*).
6. (Facultatif) Pour les Conditions (facultatif), choisissez Ajouter une condition pour créer des conditions supplémentaires à réunir avant que les utilisateurs de votre application ne puissent utiliser les autorisations que le rôle accorde. Par exemple, vous pouvez ajouter une condition qui accorde l'accès aux AWS ressources uniquement pour un ID utilisateur IAM spécifique. Vous pouvez par ailleurs ajouter des conditions à la politique de confiance après la création du rôle. Pour plus d'informations, consultez Modification d'une politique d'approbation de rôle (console).
7. Vérifiez vos informations OIDC, puis choisissez Next.
8. IAM inclut une liste des politiques AWS gérées et gérées par le client dans votre compte. Sélectionnez la politique à utiliser pour la politique d'autorisations ou choisissez Create policy (Créer une politique) pour ouvrir un nouvel onglet de navigateur et créer une nouvelle politique de bout en bout. Pour plus d'informations, consultez Création de politiques IAM. Une fois

la politique créée, fermez cet onglet et revenez à l'onglet initial. Cochez la case à côté des politiques d'autorisation que vous souhaitez imposer aux utilisateurs de l'OIDC. Si vous préférez, vous pouvez ne sélectionner aucune stratégie pour le moment, puis les attacher au rôle ultérieurement. Par défaut, un rôle ne dispose d'aucune autorisation.

9. (Facultatif) Définissez une limite d'autorisations. Il s'agit d'une fonctionnalité avancée.

   Ouvrez la section Set permissions boundary (Définir une limite d'autorisations) et choisissez Use a permissions boundary to control the maximum role permissions (Utiliser une limite d'autorisations pour contrôler le nombre maximum d'autorisations de rôle). Sélectionnez la politique à utiliser comme limite d'autorisations.
10. Choisissez Suivant.
11. Pour Role name (Nom du rôle), saisissez un nom de rôle. Les noms de rôles doivent être uniques au sein de votre Compte AWS. Ils ne dépendent pas de la casse. Par exemple, vous ne pouvez pas créer deux rôles nommés **`PRODROLE`** et **`prodrole`**. Comme d'autres AWS ressources peuvent faire référence au rôle, vous ne pouvez pas modifier le nom du rôle après l'avoir créé.
12. (Facultatif) Pour Description, saisissez une description pour le nouveau rôle.
13. Pour modifier les cas d'utilisation et les autorisations pour le rôle, choisissez Edit (Modifier) dans les sections Step 1: Select trusted entities (Étape 1 : sélection d'entités de confiance) ou Step 2: Select permissions (Étape 2 : sélection d'autorisations).
14. (Facultatif) Pour ajouter des métadonnées au rôle, attachez des balises en tant que paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Balisage des ressources IAM.
15. Passez en revue les informations du rôle, puis choisissez Créer un rôle.

Configuration d'un rôle pour le fournisseur d'identité GitHub OIDC

Si vous utilisez un GitHub fournisseur d'identité (IdP) OpenID Connect (OIDC), la meilleure pratique consiste à limiter le nombre d'entités pouvant assumer le rôle associé à l'IdP IAM. Lorsque vous incluez une déclaration de condition dans la politique de confiance, vous pouvez limiter le rôle à une GitHub organisation, un référentiel ou une branche spécifique. Vous pouvez utiliser la clé de condition `token.actions.githubusercontent.com:sub` avec des opérateurs de condition de chaîne pour limiter l'accès. Nous vous recommandons de limiter cette condition à un ensemble spécifique de référentiels ou de succursales au sein de votre GitHub organisation. Pour plus d'informations sur la façon de AWS configurer GitHub l'OIDC de confiance en tant qu'identité fédérée, consultez GitHub Docs - Configuring OpenID Connect in Amazon Web Services.

Si vous utilisez GitHub des environnements dans des workflows d'action ou dans des politiques OIDC, nous vous recommandons vivement d'ajouter des règles de protection à l'environnement pour une sécurité accrue. Utilisez les branches et les balises de déploiement pour limiter les branches et les balises qui peuvent être déployées dans l'environnement. Pour plus d'informations sur la configuration des environnements dotés de règles de protection, consultez la section Branches et balises GitHub de déploiement dans l'article Utilisation des environnements pour le déploiement.

Quand GitHub l'IdP OIDC est le principal fiable pour votre rôle, IAM vérifie la condition de la politique de confiance des rôles pour vérifier que la `token.actions.githubusercontent.com:sub` clé de condition est présente et que sa valeur n'est pas uniquement un caractère générique (* et ?) ou nul. IAM effectue cette vérification lors de la création ou de la mise à jour de la politique IAM. Si la clé de condition `token.actions.githubusercontent.com:sub` est absente ou la valeur clé ne répond pas aux critères de valeur mentionnés, la demande échouera et renverra une erreur.

> **Important**
>
> Si vous ne limitez pas la clé de condition `token.actions.githubusercontent.com:sub` à une organisation ou à un référentiel spécifique, GitHub les actions provenant d'organisations ou de référentiels indépendants de votre volonté peuvent assumer les rôles associés à l' GitHub IdP IAM dans votre compte. AWS

L'exemple de politique de confiance suivant limite l'accès à l' GitHub organisation, au référentiel et à la succursale définis. La `token.actions.githubusercontent.com:sub` valeur de la clé de condition dans l'exemple suivant est le format de valeur d'objet par défaut documenté par GitHub.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::012345678910:oidc-provider/
token.actions.githubusercontent.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
```

```
          "token.actions.githubusercontent.com:sub":
 "repo:GitHubOrg/GitHubRepo:ref:refs/heads/GitHubBranch"
        }
      }
    }
  ]
}
```

L'exemple de condition suivant limite l'accès à l' GitHub organisation et au référentiel définis, mais accorde l'accès à n'importe quelle branche du référentiel.

```
"Condition": {
  "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/GitHubRepo:*"
  }
}
```

L'exemple de condition suivant limite l'accès à n'importe quel référentiel ou branche au sein de l' GitHub organisation définie. Nous vous recommandons de limiter la clé de condition `token.actions.githubusercontent.com:sub` à une valeur spécifique qui limite l'accès aux GitHub actions au sein de votre GitHub organisation.

```
"Condition": {
  "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/*"
  }
}
```

Pour plus d'informations sur les clés de fédération OIDC disponibles pour les vérifications d'état dans les politiques, consultezClés disponibles pour la AWS fédération OIDC.

Création d'un rôle pour la fédération SAML 2.0 (console)

Vous pouvez utiliser la fédération SAML 2.0 au lieu de créer des utilisateurs IAM dans votre. Compte AWS Avec un fournisseur d'identité (IdP), vous pouvez gérer vos identités d'utilisateurs

en dehors de l'extérieur AWS et leur donner les autorisations d'accéder aux AWS ressources de votre compte. Pour plus d'informations sur la fédération et les fournisseurs d'identité, consultez Fournisseurs d'identité et fédération.

> **Note**
>
> Pour améliorer la résilience de la fédération, nous vous recommandons de configurer votre IdP et votre fédération AWS pour prendre en charge plusieurs points de terminaison de connexion SAML. Pour plus de détails, consultez l'article du blog sur la AWS sécurité Comment utiliser les points de terminaison SAML régionaux pour le basculement.

Prérequis pour la création d'un rôle pour SAML

Avant de pouvoir créer un rôle pour la fédération SAML 2.0, vous devez tout d'abord suivre les étapes requises suivantes.

Pour préparer la création d'un rôle pour la fédération SAML 2.0

1. Avant de créer un rôle pour la fédération SAML, vous devez créer un fournisseur SAML dans IAM. Pour plus d'informations, veuillez consulter Création d'un fournisseur d'identité SAML dans IAM.
2. Préparez les politiques pour le rôle que les utilisateurs authentifiés SAML 2.0 vont endosser. Comme n'importe quel rôle, celui de la fédération SAML inclut deux politiques. L'une est la politique de confiance de rôle qui spécifie la personne capable d'endosser le rôle. L'autre est la politique d'autorisations IAM qui spécifie les AWS actions et les ressources auxquelles l'utilisateur fédéré est autorisé ou non à accéder.

   Lorsque vous créez la politique de confiance pour votre rôle, vous devez utiliser trois valeurs pour vous assurer que seule votre application peut assumer le rôle :

   - Pour l'élément `Action`, utilisez l'action `sts:AssumeRoleWithSAML`.
   - Pour l'élément `Principal`, utilisez la chaîne `{"Federated":`*`ARNofIdentityProvider`*`}`. Remplacez *`ARNofIdentityProvider`* par l'ARN du fournisseur d'identité SAML que vous avez créé dans Step 1.
   - Pour l'élément `Condition`, utilisez une condition `StringEquals` pour vérifier que l'attribut `saml:aud` de la réponse SAML correspond au point de terminaison de fédération SAML pour AWS.

L'exemple suivant de politique de confiance est conçu pour un utilisateur fédéré SAML :

```
{
    "Version": "2012-10-17",
    "Statement": {
      "Effect": "Allow",
      "Action": "sts:AssumeRoleWithSAML",
      "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/PROVIDER-
NAME"},
      "Condition": {"StringEquals": {"SAML:aud": "https://signin.aws.amazon.com/
saml"}}
    }
  }
```

Remplacez l'ARN principal par l'ARN réel pour le fournisseur SAML que vous avez créé dans IAM. Il utilisera votre propre ID de compte et nom du fournisseur.

Création d'un rôle pour SAML

Après avoir exécuté les étapes prérequises, vous pouvez créer le rôle pour la fédération basée sur SAML.

Pour créer un rôle pour la fédération basée sur SAML

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
3. Choisissez le type de rôle Fédération SAML 2.0.
4. Pour Select a SAML provider (Sélectionnez un fournisseur SAML), choisissez le fournisseur de votre rôle.
5. Choisissez la méthode de niveau d'accès SAML 2.0.
   - Choisissez Autoriser l'accès par programmation uniquement pour créer un rôle pouvant être assumé par programmation depuis l' AWS API ou. AWS CLI
   - Choisissez Autoriser la programmation et AWS Management Console l'accès pour créer un rôle qui peut être assumé par programmation et à partir du. AWS Management Console

Les rôles créés par les deux méthodes sont similaires, mais le rôle qui peut aussi être endossé depuis la console inclut une politique d'approbation contenant une condition particulière. Cette condition vérifie explicitement que l'audience SAML (l'attribut `SAML:aud`) est définie sur le point de terminaison de connexion AWS pour SAML (https://signin.aws.amazon.com/saml).

6. Si vous créez un rôle pour un accès par programme, choisissez un attribut dans la liste Attribut. Ensuite, dans le champ Value (Valeur), tapez une valeur à inclure dans le rôle. Cette valeur restreint l'accès au rôle aux utilisateurs du fournisseur d'identité dont la réponse d'authentification SAML (assertion) inclut les attributs que vous spécifiez. Vous devez spécifier au moins un attribut afin de garantir la limitation du rôle à un sous-ensemble d'utilisateurs de votre organisation.

   Si vous créez un rôle pour l'accès par programmation et via la console, l'attribut `SAML:aud` est automatiquement ajouté et sa valeur est définie sur l'URL du point de terminaison SAML AWS (https://signin.aws.amazon.com/saml).

7. Pour ajouter d'autres conditions liées aux attributs à la politique d'approbation, choisissez Condition (optional) (Conditions [facultatif]), sélectionnez la condition supplémentaire et spécifiez une valeur.

   > **Note**
   >
   > La liste inclut les attributs SAML les plus couramment utilisés. IAM prend en charge des attributs supplémentaires que vous pouvez utiliser pour créer des conditions. Pour obtenir la liste des attributs pris en charge, veuillez consulter Clés disponibles pour la fédération SAML. Si vous avez besoin d'une condition pour un attribut SAML pris en charge ne figurant pas dans la liste, vous pouvez ajouter cette condition manuellement. Pour ce faire, modifiez la politique de confiance après la création du rôle.

8. Passez en revue les informations d'approbation SAML 2.0, puis choisissez Next: Permissions (Suivant : Autorisations).

9. IAM inclut une liste des politiques AWS gérées et gérées par le client dans votre compte. Sélectionnez la politique à utiliser pour la politique d'autorisations ou choisissez Create policy (Créer une politique) pour ouvrir un nouvel onglet de navigateur et créer une nouvelle politique de bout en bout. Pour plus d'informations, consultez Création de politiques IAM. Une fois la politique créée, fermez cet onglet et revenez à l'onglet initial. Cochez la case à côté des politiques d'autorisation que vous souhaitez appliquer aux utilisateurs fédérés OIDC. Si vous

préférez, vous pouvez ne sélectionner aucune stratégie pour le moment, puis les attacher au rôle ultérieurement. Par défaut, un rôle ne dispose d'aucune autorisation.

10. (Facultatif) Définissez une limite d'autorisations. Il s'agit d'une fonctionnalité avancée.

    Ouvrez la section Set permissions boundary (Définir une limite d'autorisations) et choisissez Use a permissions boundary to control the maximum role permissions (Utiliser une limite d'autorisations pour contrôler le nombre maximum d'autorisations de rôle). Sélectionnez la politique à utiliser comme limite d'autorisations.
11. Choisissez Suivant.
12. Choisissez Suivant : vérification.
13. Pour Role name (Nom du rôle), saisissez un nom de rôle. Les noms de rôles doivent être uniques au sein de votre Compte AWS. Ils ne sont pas sensibles à la casse. Par exemple, vous ne pouvez pas créer deux rôles nommés **`PRODROLE`** et **`prodrole`**. Dans la mesure AWS où d'autres ressources peuvent faire référence au rôle, vous ne pouvez pas modifier le nom du rôle une fois celui-ci créé.
14. (Facultatif) Pour Description, saisissez une description pour le nouveau rôle.
15. Choisissez Edit (Modifier) dans les sections Step 1: Select trusted entities (Étape 1 : sélection d'entités de confiance) ou Step 2: Add permissions (Étape 2 : ajouter des autorisations) pour modifier les cas d'utilisation et les autorisations pour le rôle.
16. (Facultatif) Ajoutez des métadonnées au rôle en associant les identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Balisage des ressources IAM.
17. Passez en revue les informations du rôle, puis choisissez Créer un rôle.

Après avoir créé le rôle, vous devez finaliser la relation d'approbation SAML en configurant le logiciel de votre fournisseur d'identité à l'aide d'informations sur AWS. Ces informations incluent les rôles que vous souhaitez que vos utilisateurs fédérés utilisent. Il s'agit de la configuration de la relation d'approbation des parties utilisatrices entre votre fournisseur d'identité et AWS. Pour plus d'informations, voir Configurez votre IdP SAML 2.0 en vous fiant à la confiance des parties et en ajoutant des réclamations.

Création d'un rôle à l'aide de politiques d'approbation personnalisées (console)

Vous pouvez créer une politique de confiance personnalisée pour déléguer l'accès et permettre à d'autres personnes d'effectuer des actions dans votre Compte AWS. Pour plus d'informations, consultez Création de politiques IAM.

Pour plus d'informations sur la façon d'utiliser les rôles pour déléguer des autorisations, consultez Termes et concepts relatifs aux rôles.

Création d'un rôle IAM à l'aide d'une politique d'approbation personnalisée (console)

Vous pouvez utiliser le AWS Management Console pour créer un rôle qu'un utilisateur IAM peut assumer. Supposons, par exemple, que votre organisation en dispose Comptes AWS de plusieurs pour isoler un environnement de développement d'un environnement de production. Pour des informations générales sur la création d'un rôle autorisant les utilisateurs du compte de développement à accéder aux ressources du compte de production, consultez Exemple de scénario utilisant des comptes de développement et de production distincts.

Pour créer un rôle à l'aide d'une politique d'approbation personnalisée (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console, choisissez Rôles, puis Créer un rôle.
3. Choisissez le type de rôle Custom trust policy (Politique d'approbation personnalisée).
4. Dans la section Custom trust policy (Politique d'approbation personnalisée), saisissez ou collez la politique d'approbation personnalisée pour le rôle. Pour plus d'informations, consultez Création de politiques IAM.
5. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Next (Suivant).
6. Activez la case à cocher à côté de la politique d'approbation que vous avez créée.
7. (Facultatif) Définissez une limite d'autorisations. Il s'agit d'une fonctionnalité avancée disponible pour les fonctions de service, mais pas pour les rôles liés à un service.

   Ouvrez la section Set permissions boundary (Définir une limite d'autorisations) et choisissez Use a permissions boundary to control the maximum role permissions (Utiliser une limite d'autorisations pour contrôler le nombre maximum d'autorisations de rôle). IAM inclut une liste des politiques AWS gérées et gérées par le client dans votre compte. Sélectionnez la politique à utiliser comme limite d'autorisations.
8. Choisissez Suivant.
9. Pour Nom du rôle, le degré de la personnalisation du nom du rôle est défini par le service. Si le service définit le nom du rôle, cette option n'est pas modifiable. Dans d'autres cas, le service peut définir un préfixe pour le rôle ou vous permettre de saisir un suffixe facultatif. Certains services vous permettent de spécifier le nom complet de votre rôle.

Si possible, saisissez un nom de rôle ou un suffixe de nom de rôle. Les noms de rôles doivent être uniques au sein de votre Compte AWS. Ils ne sont pas sensibles à la casse. Par exemple, vous ne pouvez pas créer deux rôles nommés **`PRODROLE`** et **`prodrole`**. Comme d'autres AWS ressources peuvent faire référence au rôle, vous ne pouvez pas modifier le nom du rôle une fois celui-ci créé.

10. (Facultatif) Pour Description, saisissez une description pour le nouveau rôle.
11. Choisissez Edit (Modifier) dans les sections Step 1: Select trusted entities (Étape 1 : sélection d'entités de confiance) ou Step 2: Add permissions (Étape 2 : Ajouter des autorisations) pour modifier la politique et les autorisations personnalisées pour le rôle.
12. (Facultatif) Ajoutez des métadonnées au rôle en associant les identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Balisage des ressources IAM.
13. Passez en revue les informations du rôle, puis choisissez Créer un rôle.

Exemples de politiques pour la délégation d'accès

Les exemples suivants montrent comment vous pouvez autoriser ou accorder l' Compte AWS accès aux ressources d'un autre Compte AWS. Pour apprendre à créer une politique IAM à l'aide de ces exemples de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

Rubriques

- Utiliser des rôles pour déléguer l'accès aux ressources d'une autre Compte AWS ressource
- Utilisation d'une politique pour la délégation d'accès à des services
- Utilisation d'une politique basée sur les ressources pour la délégation de l'accès à un compartiment Amazon S3 dans un autre compte
- Utilisation d'une politique basée sur les ressources pour la délégation de l'accès à une file d'attente Amazon SQS dans un autre compte
- Délégation d'accès impossible lorsque l'accès est refusé au compte

Utiliser des rôles pour déléguer l'accès aux ressources d'une autre Compte AWS ressource

Vous trouverez un didacticiel qui montre comment utiliser des rôles IAM pour accorder aux utilisateurs d'un compte l'accès aux ressources AWS d'un autre compte ici : Didacticiel IAM : déléguer l'accès entre des comptes AWS à l'aide des rôles IAM.

> **Important**
>
> Vous pouvez inclure l'ARN d'un rôle ou d'un utilisateur dans l'élément `Principal` de la politique d'approbation d'un rôle. Lorsque vous enregistrez la politique, AWS transforme l'ARN en un identifiant principal unique. Cela permet de réduire le risque d'escalade des privilèges par la suppression et la nouvelle création du rôle ou de l'utilisateur. Cet ID n'est pas fréquent dans la console, car il existe également une transformation inverse, pour revenir à l'ARN, lorsque la politique d'approbation est affichée. Cependant, si vous supprimez le rôle ou l'utilisateur, la relation est interrompue. La politique ne s'applique plus, même si vous recréez l'utilisateur ou le rôle, étant donné qu'il ne correspond pas à l'ID du principal stocké dans la politique d'approbation. Lorsque cela se produit, l'ID principal apparaît dans la console car il n'est plus AWS possible de le mapper à un ARN. Le résultat final est que si vous supprimez et recréez un utilisateur ou un rôle référencé dans l'élément `Principal` d'une politique d'approbation, vous devez modifier le rôle afin de remplacer l'ARN. Il deviendra le nouvel ID du principal lorsque vous enregistrez la politique.

Utilisation d'une politique pour la délégation d'accès à des services

L'exemple suivant illustre une politique qui est peut être attachée à un rôle. La politique permet à deux services, Amazon EMR et Amazon AWS Data Pipeline, d'assumer ce rôle. Les services peuvent ensuite effectuer toutes les tâches autorisées par la politique d'autorisation affectée au rôle (non illustré). Pour définir plusieurs principaux de service, vous ne spécifiez pas deux éléments `Service` ; vous ne devez en avoir qu'un seul. À la place, vous utilisez un tableau de plusieurs principaux de service comme valeur d'un élément `Service` unique.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
```

```
          "elasticmapreduce.amazonaws.com",
          "datapipeline.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Utilisation d'une politique basée sur les ressources pour la délégation de l'accès à un compartiment Amazon S3 dans un autre compte

Dans cet exemple, le compte A utilise une politique basée sur les ressources (une politique de compartiment Amazon S3) pour octroyer au compte B l'accès complet au compartiment S3 du compte A. Ensuite, le compte B crée une politique d'utilisateur IAM pour déléguer l'accès au compartiment du compte A à l'un des utilisateurs du compte B.

La politique de compartiment S3 du compte A peut se présenter comme suit. Dans cet exemple, le compartiment S3 du compte A est nommé mybucket, et le numéro de compte du compte B est 111122223333. Il ne spécifie pas d'utilisateurs ou de groupes dans le compte B, simplement le compte proprement dit.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "AccountBAccess1",
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }
}
```

En variante, le compte A peut utiliser des Listes de contrôle d'accès (ACL) Amazon S3 pour accorder l'accès au compte B à un compartiment S3 ou à un seul objet dans un compartiment. Dans ce cas, seule change la façon dont le compte A accorde l'accès au compte B. Le compte B utilise toujours une politique pour déléguer l'accès à un groupe IAM dans le compte B, comme décrit dans la dernière partie de cet exemple. Pour plus d'informations sur le contrôle de l'accès aux compartiments

et aux objets S3, veuillez consulter contrôle des accès dans le guide de l'utilisateur du service de stockage simple Amazon.

L'administrateur du compte B peut créer l'exemple de politique suivant. La politique donne un accès en lecture à un groupe ou à un utilisateur au sein du compte B. La politique précédente accorde l'accès au compte B. Toutefois, des groupes et des utilisateurs spécifiques au sein du compte B ne peuvent pas accéder à la ressource tant qu'une politique de groupe ou d'utilisateur ne leur accorde pas explicitement des autorisations pour cette ressource. Les autorisations de cette politique peuvent uniquement être un sous-ensemble de celles de la politique intercompte précédente. Le compte B ne peut pas accorder plus d'autorisations à ses utilisateurs et groupes que le compte A ne lui a accordé dans la première politique. Dans cette politique, l'élément `Action` est défini explicitement pour autoriser uniquement les actions `List`, tandis que l'élément `Resource` de cette politique correspond à l'élément `Resource` pour la politique de compartiment implémentée par le compte A.

Pour implémenter cette politique, le compte B utilise IAM pour l'attacher à l'utilisateur (ou au groupe) approprié dans le compte B.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:List*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }
}
```

Utilisation d'une politique basée sur les ressources pour la délégation de l'accès à une file d'attente Amazon SQS dans un autre compte

Dans l'exemple suivant, le compte A est doté d'une file d'attente Amazon SQS qui utilise une politique basée sur les ressources attachée à celle-ci pour accorder au compte B l'accès à la file d'attente. Le compte B utilise ensuite une politique de groupe IAM pour déléguer l'accès à un groupe du compte B.

L'exemple de politique de file d'attente suivant accorde au compte B l'autorisation d'effectuer les actions `SendMessage` et `ReceiveMessage` dans la file d'attente du compte A nommée queue1, mais uniquement entre midi et 15 heures le 30 novembre 2014. Le numéro de compte du compte B est 1111-2222-3333. Le compte A utilise Amazon SQS pour implémenter cette politique.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": [
      "sqs:SendMessage",
      "sqs:ReceiveMessage"
    ],
    "Resource": ["arn:aws:sqs:*:123456789012:queue1"],
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2014-11-30T12:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2014-11-30T15:00Z"}
    }
  }
}
```

La politique du compte B pour la délégation d'accès à un groupe dans le compte B peut se présenter comme l'exemple suivant. Le compte B utilise IAM pour attacher cette politique à un groupe (ou utilisateur).

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sqs:*",
    "Resource": "arn:aws:sqs:*:123456789012:queue1"
  }
}
```

Dans l'exemple de politique utilisateur IAM précédent, le compte B utilise un caractère générique pour accorder à son utilisateur l'accès à toutes les actions Amazon SQS dans la file d'attente du compte A. Cependant, le compte B peut déléguer l'accès dans la mesure où celui-ci lui a été accordé. Le groupe du compte B qui a une seconde politique peut accéder à la file d'attente uniquement entre midi et 15 heures le 30 novembre 2014. L'utilisateur peut uniquement effectuer les actions `SendMessage` et `ReceiveMessage`, tel que défini dans la politique de file d'attente Amazon SQS du compte A.

Délégation d'accès impossible lorsque l'accès est refusé au compte

Compte AWS On ne peut pas déléguer l'accès aux ressources d'un autre compte si celui-ci a explicitement refusé l'accès au compte parent de l'utilisateur. Le rejet se propage aux utilisateurs sous ce compte, qu'ils aient ou non des politiques existantes leur accordant l'accès.

Par exemple, le compte A crée une politique de compartiment pour son compartiment S3 qui refuse explicitement au compte B l'accès à ce compartiment. Mais le compte B crée une politique d'utilisateur IAM qui accorde à un utilisateur du compte B l'accès au compartiment du compte A. Le refus explicite appliqué au compartiment S3 du compte A est propagé aux utilisateurs du compte B. Il remplace la politique d'utilisateur IAM qui accorde l'accès aux utilisateurs du compte B. (Pour des informations détaillées sur la façon dont les autorisations sont évaluées, consultez Logique d'évaluation de politiques.)

La politique de compartiment du compte A peut se présenter comme suit. Dans cet exemple, le compartiment S3 du compte A est nommé mybucket, et le numéro de compte du compte B est 1111-2222-3333. Le compte A utilise Amazon S3 pour implémenter cette politique.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "AccountBDeny",
    "Effect": "Deny",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::mybucket/*"
  }
}
```

Ce refus explicite remplace toutes les politiques au sein du compte B qui accordent l'accès au compartiment S3 du compte A.

Utilisation de rôles IAM

Avant qu'un utilisateur, une application ou un service soit en mesure d'utiliser un rôle que vous avez créé, vous devez lui accorder les autorisations nécessaires pour changer de rôle. Vous pouvez utiliser n'importe quelle politique attachée aux groupes ou aux utilisateurs pour accorder les autorisations nécessaires. Cette section décrit l'octroi d'autorisations aux utilisateurs pour utiliser un rôle. Il explique également comment l'utilisateur peut passer à un rôle depuis les AWS

Management Console Outils pour Windows PowerShell, le AWS Command Line Interface (AWS CLI) et l'`AssumeRole`API.

> **Important**
>
> Si vous créez un rôle par programmation plutôt que dans la console IAM, vous avez la possibilité d'ajouter un chemin `Path` de 512 caractères au maximum à l'élément `RoleName` qui lui, est de 64 caractères au maximum. Toutefois, si vous avez l'intention d'utiliser un rôle avec la fonctionnalité Switch Role dans le AWS Management Console, la combinaison `Path` `RoleName` ne peut pas dépasser 64 caractères.

Vous pouvez changer de rôle depuis le AWS Management Console. Vous pouvez assumer un rôle en appelant une opération d'API AWS CLI ou en utilisant une URL personnalisée. La méthode que vous utilisez détermine qui peut endosser le rôle et la durée de la session de rôle. Lorsque vous l'utilisez `AssumeRole*` les opérations d'API, le rôle IAM que vous endossez est la ressource. Le rôle ou l'utilisateur IAM qui appelle les opérations de l'API `AssumeRole*` est le principal.

Comparaison des méthodes pour l'utilisation de rôles

Méthode permettant d'endosser le rôle	Qui peut endosser le rôle	Méthode permettant de spécifier la durée de vie des informations d'identification	Durée de vie des informations d'identification (min | max | par défaut)
AWS Management Console	Utilisateur (en changeant de rôles)	Durée de session maximale sur la page récapitulative des rôles	15 min | Durée de session maximale[2] | 1 h
Opération d'interface de ligne de commande (CLI) `assume-ro`	Utilisateur ou rôle[1]	Paramètre de CLI `duration-seconds` ou d'API	15 min | Durée de session maximale[2] | 1 h

Méthode permettant d'endosser le rôle	Qui peut endosser le rôle	Méthode permettant de spécifier la durée de vie des informations d'identification	Durée de vie des informations d'identification (min | max | par défaut)
le ou d'API AssumeRole		DurationSeconds	
Opération d'interface de ligne de commande (CLI) assume-role-with-saml ou d'API AssumeRoleWithSAML	Tout utilisateur authentifié utilisant SAML	Paramètre de CLI duration-seconds ou d'API DurationSeconds	15 min | Durée de session maximale² | 1 h
Opération d'interface de ligne de commande (CLI) assume-role-with-web-identity ou d'API AssumeRoleWithWebIdentity	Tout utilisateur authentifié à l'aide d'un fournisseur OIDC	Paramètre de CLI duration-seconds ou d'API DurationSeconds	15 min | Durée de session maximale² | 1 h
URL de console construite avec AssumeRole	Utilisateur ou rôle	Paramètre HTML SessionDuration dans l'URL	15 min | 12 h | 1 h

Méthode permettant d'endosser le rôle	Qui peut endosser le rôle	Méthode permettant de spécifier la durée de vie des informations d'identification	Durée de vie des informations d'identification (min | max | par défaut)
URL de console construite avec `AssumeRoleWithSAML`	Tout utilisateur authentifié utilisant SAML	Paramètre HTML `SessionDuration` dans l'URL	15 min | 12 h | 1 h
URL de console construite avec `AssumeRoleWithWebIdentity`	Tout utilisateur authentifié à l'aide d'un fournisseur OIDC	Paramètre HTML `SessionDuration` dans l'URL	15 min | 12 h | 1 h

[1] L'utilisation des informations d'identification pour qu'un rôle endosse un rôle différent est appelée création de chaînes de rôles. Lorsque vous utilisez la création de chaînes de rôles, vos nouvelles informations d'identification sont limitées à une durée maximale d'une heure. Lorsque vous utilisez des rôles pour accorder des autorisations aux applications qui s'exécutent sur des instances EC2, ces applications ne sont pas soumises à cette limitation.

[2] La valeur de ce paramètre peut varier de 1 heure à 12 heures. Pour en savoir plus sur la modification du paramètre de durée de session maximale, consultez Modification d'un rôle. Ce paramètre détermine la durée de session maximale que vous pouvez demander lorsque vous obtenez les informations d'identification du rôle. Par exemple, lorsque vous utilisez les opérations d'API AssumeRole* pour assumer un rôle, vous pouvez spécifier une durée de session à l'aide du `DurationSeconds` paramètre. Utilisez ce paramètre pour spécifier la durée de la session du rôle entre 900 secondes (15 minutes) et la durée de session maximale pour le rôle. Les utilisateurs IAM qui changent de rôle dans la console se voient accorder la durée de session maximale ou le temps restant dans la session de l'utilisateur, selon la durée la plus courte. Supposons que vous définissiez une durée maximale de 5 heures sur un rôle. Un utilisateur IAM qui s'est connecté à la console pendant 10 heures (sur 12, le maximum par défaut) décide d'endosser le rôle. La durée de la session de rôle disponible est de 2 heures. Pour savoir comment afficher la valeur maximale pour votre rôle,

consultez Affichage du paramètre de durée de session maximale pour un rôle plus loin sur cette page.

Remarques

- Le paramètre de durée maximale de session ne limite pas les sessions endossées par les services AWS .
- Les informations d'identification du rôle Amazon EC2 IAM ne sont pas soumises aux durées de session maximales configurées dans le rôle.
- Pour permettre aux utilisateurs d'assumer à nouveau le rôle actuel au cours d'une session de rôle, spécifiez l'ARN ou l' Compte AWS ARN comme principal dans la politique de confiance des rôles. Services AWS qui fournissent des ressources de calcul telles qu'Amazon EC2, Amazon ECS, Amazon EKS et Lambda fournissent des informations d'identification temporaires et mettent automatiquement à jour ces informations d'identification. Cela garantit que vous disposez toujours d'un ensemble d'informations d'identification valide. Pour ces services, il n'est pas nécessaire d'endosser à nouveau le rôle actuel pour obtenir des informations d'identification temporaires. Toutefois, si vous avez l'intention de transférer des balises de session ou une politique de session, vous devez endosser à nouveau le rôle actuel. Pour savoir comment modifier une politique d'approbation des rôles afin d'ajouter l'ARN ou Compte AWS l'ARN du rôle principal, consultezModification d'une politique d'approbation de rôle (console).

Rubriques

- Affichage du paramètre de durée de session maximale pour un rôle
- Octroi d'autorisations à un utilisateur pour endosser un rôle
- Octroi d'autorisations à un utilisateur pour transférer un rôle à un service AWS
- Changement de rôle (console)
- Assumer un rôle IAM (AWS CLI)
- Passer à un rôle IAM (Outils pour Windows PowerShell)
- Passage à un rôle IAM (AWS API)
- Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2
- Révocation des informations d'identification de sécurité temporaires d'un rôle IAM

Affichage du paramètre de durée de session maximale pour un rôle

Vous pouvez spécifier la durée maximale de session pour un rôle à l' AWS Management Console aide de l' AWS API AWS CLI or ou. Lorsque vous utilisez une opération AWS CLI ou une API pour assumer un rôle, vous pouvez spécifier une valeur pour le `DurationSeconds` paramètre. Vous pouvez utiliser ce paramètre pour spécifier la durée de session du rôle, entre 900 secondes (15 minutes) et la durée de session maximale définie pour le rôle. Avant de spécifier le paramètre, vous devez consulter ce paramètre pour votre rôle. Si vous spécifiez une valeur supérieure à la valeur maximale pour le paramètre `DurationSeconds`, l'opération échoue.

Pour afficher la durée de session maximale d'un rôle (console)

1. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles).
2. Choisissez le nom du rôle que vous souhaitez consulter.
3. À côté de Maximum session duration (Durée de session maximale), affichez la longueur de session maximale octroyée pour le rôle. Il s'agit de la durée maximale de session que vous pouvez spécifier dans votre AWS CLI opération ou dans celle de l'API.

Pour afficher le paramètre de durée de session maximale d'un rôle (AWS CLI)

1. Si vous ne connaissez pas le nom du rôle que vous souhaitez endosser, exécutez la commande suivante pour répertorier les rôles dans votre compte :
   - aws iam list-roles
2. Pour afficher la durée de session maximale du rôle, exécutez la commande suivante. Consultez ensuite le paramètre de durée de session maximale.
   - aws iam get-role

Pour consulter le paramètre de durée maximale de session (AWS API) d'un rôle

1. Si vous ne connaissez pas le nom du rôle que vous souhaitez endosser, appelez l'opération suivante pour répertorier les rôles dans votre compte :
   - ListRoles
2. Pour afficher la durée de session maximale du rôle, appelez l'opération suivante. Consultez ensuite le paramètre de durée de session maximale.

- GetRole

Octroi d'autorisations à un utilisateur pour endosser un rôle

Lorsqu'un administrateur crée un rôle pour un accès intercompte, il établit une relation d'approbation entre le compte propriétaire du rôle et des ressources (compte d'approbation) et le compte qui contient les utilisateurs (compte approuvé). Pour cela, l'administrateur du compte d'approbation spécifie le numéro du compte approuvé en tant que `Principal` dans la politique d'approbation du rôle. Cela permet potentiellement à n'importe quel utilisateur du compte approuvé d'endosser le rôle. Pour achever la configuration, l'administrateur du compte approuvé doit accorder l'autorisation d'endosser ce rôle à des groupes ou utilisateurs spécifiques du compte.

Pour accorder l'autorisation de passer à un rôle

1. En tant qu'administrateur du compte approuvé, créez une nouvelle politique pour l'utilisateur ou modifiez une politique existante pour y ajouter les éléments requis. Pour plus de détails, consultez Création ou modification de la politique.
2. Choisissez ensuite la manière dont vous souhaitez partager les informations du rôle :

   - Lien du rôle : envoyez aux utilisateurs un lien qui les dirige vers la page Switch Role (Changer de rôle) dans laquelle tous les détails sont déjà remplis.
   - ID ou alias du compte : donnez à chaque utilisateur le nom du rôle ainsi que l'ID ou l'alias du compte. L'utilisateur accède ensuite à la page Switch Role (Changer de rôle) et ajoute les détails manuellement.

   Pour plus de détails, consultez Fourniture d'informations à l'utilisateur.

Notez que vous ne pouvez changer de rôle que lorsque vous vous connectez en tant qu'utilisateur IAM, en tant que rôle fédéré SAML ou en tant que rôle fédéré d'identité Web. Vous ne pouvez pas changer de rôle lorsque vous vous connectez en tant qu' Utilisateur racine d'un compte AWS.

Important

Vous ne pouvez pas changer AWS Management Console de rôle dans un rôle nécessitant une ExternalIdvaleur. Vous ne pouvez basculer vers un tel rôle qu'en appelant l'API `AssumeRole` qui prend en charge le paramètre `ExternalId`.

Remarques

- Cette rubrique décrit les politiques applicables aux utilisateurs car, en fin de compte, il s'agit ici d'accorder à un utilisateur les autorisations nécessaires pour effectuer une tâche. Cependant, nous vous déconseillons d'octroyer des autorisations directement à un utilisateur individuel. Lorsqu'un utilisateur assume un rôle, il reçoit également les autorisations associées.
- Lorsque vous changez de rôle dans le AWS Management Console, la console utilise toujours vos informations d'identification d'origine pour autoriser le changement. Cela s'applique que vous soyez connecté en tant qu'utilisateur IAM, en tant que rôle fédéré SAML ou en tant que rôle fédéré d'identité web. Par exemple, si vous basculez sur RoleA, IAM utilise les informations d'identification du compte utilisateur ou du rôle fédéré d'origine pour déterminer si vous êtes autorisé à assumer le rôle RoleA. Si vous essayez ensuite de basculer sur RoleB pendant que vous utilisez RoleA, vos informations d'identification d'utilisateur ou du rôle fédéré d'origine sont utilisées pour autoriser votre tentative. Les informations d'identification de RoleA (Rôle A) ne sont pas utilisées pour cette action.

Rubriques

- Création ou modification de la politique
- Fourniture d'informations à l'utilisateur

Création ou modification de la politique

Une politique qui accorde à un utilisateur l'autorisation d'endosser un rôle doit inclure une instruction avec l'effet `Allow` sur les éléments suivants :

- L'action `sts:AssumeRole`
- L'Amazon Resource Name (ARN) du rôle dans un élément `Resource`

Les utilisateurs bénéficiant de la politique sont autorisés à endosser les rôles figurant sur la ressource (via l'appartenance à un groupe ou directement).

> **Note**
>
> Si `Resource` est définie sur `*`, l'utilisateur peut endosser n'importe quel rôle dans n'importe quel compte faisant confiance au compte de l'utilisateur. (En d'autres termes, la politique de confiance du rôle spécifie le compte de l'utilisateur en tant que `Principal`). La bonne pratique consiste à appliquer le principe du moindre privilège et à spécifier l'ARN complet uniquement pour les rôles dont l'utilisateur a besoin.

L'exemple suivant illustre une politique qui permet à l'utilisateur d'endosser des rôles dans un seul compte. En outre, la politique utilise un caractère générique (*) pour spécifier que l'utilisateur ne peut passer à un rôle que si le nom du rôle commence par les lettres `Test`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account-id:role/Test*"
  }
}
```

> **Note**
>
> Les autorisations que le rôle octroie à l'utilisateur ne viennent pas s'ajouter aux autorisations dont il dispose déjà. Lorsqu'un utilisateur endosse un rôle, il abandonne temporairement ses autorisations d'origine, de manière à adopter celles accordées par le rôle. Lorsqu'il quitte le rôle, ses autorisations d'origine sont automatiquement restaurées. Par exemple, supposons que les autorisations de l'utilisateur lui permettent d'utiliser des instances Amazon EC2, mais que la politique des autorisations du rôle n'accorde pas ces autorisations. Dans ce cas, lorsque vous utilisez le rôle, l'utilisateur peut ne pas utiliser les instances Amazon EC2 dans la console. En outre, les informations d'identification temporaires obtenues via `AssumeRole` ne fonctionnent pas par programmation avec les instances Amazon EC2.

Fourniture d'informations à l'utilisateur

Une fois que vous avez créé un rôle et accordé à l'utilisateur les autorisations requises pour l'endosser, vous devez également fournir à l'utilisateur les éléments suivants :

- Nom du rôle
- ID ou alias du compte qui contient le rôle

Pour faciliter l'accès aux utilisateurs, vous pouvez leur envoyer un lien préconfiguré contenant l'ID du compte et le nom du rôle. Vous pouvez voir le lien du rôle après avoir terminé l'assistant Créer un rôle en sélectionnant la bannière Afficher le rôle ou sur la page Résumé du rôle pour n'importe quel rôle inter-comptes.

Vous pouvez également utiliser le format suivant pour créer le lien manuellement. Remplacez les deux paramètres de l'exemple suivant par l'ID ou l'alias de votre compte et le nom du rôle.

```
https://signin.aws.amazon.com/switchrole?
account=your_account_ID_or_alias&roleName=optional_path/role_name
```

Nous vous recommandons de diriger vos utilisateurs vers la rubrique Changement de rôle (console) pour les guider à travers le processus. Pour résoudre les problèmes courants que vous pouvez rencontrer lorsque vous endossez un rôle, consultez Je ne parviens pas à endosser un rôle.

Considérations

- Si vous créez le rôle par programmation, vous pouvez définir un chemin et un nom. Dans ce cas, vous devez fournir le chemin complet et le nom du rôle à vos utilisateurs afin qu'ils les saisissent sur la page Switch Role (Changer de rôle) de la AWS Management Console. Par exemple : `division_abc/subdivision_efg/role_XYZ`.
- Si vous créez le rôle par programmation, vous pouvez ajouter un `Path` de 512 caractères au maximum , ainsi qu'un `RoleName`. Le nom du rôle peut comporter jusqu'à 64 caractères. Toutefois, pour utiliser un rôle avec la fonctionnalité Switch Role dans le AWS Management Console, la combinaison `Path` `RoleName` ne peut pas dépasser 64 caractères.

- Pour des raisons de sécurité, vous pouvez consulter AWS CloudTrail les journaux pour savoir qui a effectué une action dans AWS. Vous pouvez utiliser la clé de condition `sts:SourceIdentity` dans la politique d'approbation de rôle pour exiger des utilisateurs qu'ils spécifient une identité lorsqu'ils endossent un rôle. Par exemple, vous pouvez exiger que les utilisateurs IAM spécifient

leur propre nom d'utilisateur comme identité de source. Cela peut vous aider à déterminer quel utilisateur a effectué une action spécifique dans AWS. Pour plus d'informations, consultez `sts:SourceIdentity`. Vous pouvez utiliser `sts:RoleSessionName` pour exiger des utilisateurs qu'ils spécifient un nom de session lorsqu'ils endossent un rôle. Cela peut vous aider à différencier les sessions de rôle lorsqu'un rôle est utilisé par différents principaux.

Octroi d'autorisations à un utilisateur pour transférer un rôle à un service AWS

Pour configurer de nombreux AWS services, vous devez leur transmettre un rôle IAM. Cela autorise le service à assumer ultérieurement le rôle et à effectuer des actions en votre nom. Pour la plupart des services, il vous suffit de transférer le rôle au service une fois au cours de la configuration, et non chaque fois que le service assume le rôle. Par exemple, supposons que vous avez une application s'exécutant sur une instance Amazon EC2. Cette application a besoin d'informations d'identification temporaires pour l'authentification et d'autorisations pour autoriser l'application à exécuter des actions dans AWS. Lorsque vous configurez l'application, vous devez transmettre un rôle à Amazon EC2 à utiliser avec l'instance qui fournit ces informations d'identification. Vous définissez les autorisations pour les applications s'exécutant sur l'instance en attachant une politique IAM au rôle. L'application endosse le rôle chaque fois qu'elle doit effectuer les actions qui sont autorisées par le rôle.

Pour transmettre un rôle (et ses autorisations) à un AWS service, un utilisateur doit disposer des autorisations nécessaires pour transmettre le rôle au service. Cela permet aux administrateurs de s'assurer que seuls les utilisateurs autorisés peuvent configurer un service avec un rôle qui accorde des autorisations. Pour permettre à un utilisateur de transmettre un rôle à un AWS service, vous devez accorder l'`PassRole`autorisation à l'utilisateur, au rôle ou au groupe IAM de l'utilisateur.

⚠ Warning

- Vous ne pouvez utiliser cette `PassRole` autorisation que pour transmettre un rôle IAM à un service qui partage le même AWS compte. Pour transmettre un rôle du compte A à un service du compte B, vous devez d'abord créer un rôle IAM dans le compte B qui peut assumer le rôle du compte A, puis le rôle du compte B peut être transmis au service. Pour plus de détails, consultez Accès intercompte aux ressources dans IAM.
- N'essayez pas de contrôler les personnes susceptibles de faire passer un rôle en étiquetant ce rôle, puis en utilisant la clé de condition `ResourceTag` dans une politique avec l'action `iam:PassRole`. Cette approche ne produit pas des résultats fiables.

Lorsque vous définissez l'`PassRole`autorisation, vous devez vous assurer qu'un utilisateur ne transfère pas un rôle pour lequel le rôle dispose de plus d'autorisations que vous ne le souhaitez. Par exemple, Alice n'est peut-être pas autorisée à effectuer des actions Amazon S3. Si Alice pouvait transmettre un rôle à un service qui autorise les actions Amazon S3, le service pourrait effectuer des actions Amazon S3 pour le compte d'Alice lors de l'exécution de la tâche.

Lorsque vous spécifiez un rôle lié à un service, vous devez également posséder l'autorisation de transmettre ce rôle au service. Certains services créent automatiquement un rôle lié à un service dans votre compte lorsque vous effectuez une action dans ce service. Par exemple, Amazon EC2 Auto Scaling crée automatiquement le rôle lié au service `AWSServiceRoleForAutoScaling` la première fois que vous créez un groupe Auto Scaling. Si vous essayez de spécifier le rôle lié au service lorsque vous créez un groupe Auto Scaling sans l'autorisation `iam:PassRole`, vous recevez une erreur. Si vous ne spécifiez pas explicitement le rôle, l'autorisation `iam:PassRole` n'est pas requise et la valeur par défaut est d'utiliser le rôle `AWSServiceRoleForAutoScaling` pour toutes les opérations effectuées sur ce groupe. Pour savoir quels services prennent en charge les rôles liés à un service, veuillez consulter AWS services qui fonctionnent avec IAM. Pour savoir quels services créent automatiquement un rôle lié à un service lorsque vous effectuez une action dans ce service, choisissez le lien Oui et consultez la documentation relative au rôle lié à un service pour le service.

Un utilisateur peut transmettre un ARN de rôle comme paramètre dans n'importe quelle opération d'API qui utilise le rôle pour attribuer des autorisations au service. Le service vérifie ensuite si cet utilisateur dispose de l'autorisation `iam:PassRole`. Pour que l'utilisateur transfère uniquement les rôles approuvés, vous pouvez filtrer l'autorisation `iam:PassRole` à l'aide de l'élément `Resources` de l'instruction de politique IAM.

Vous pouvez utiliser `Condition` cet élément dans une politique JSON pour tester la valeur des clés incluses dans le contexte de toutes les AWS demandes. Pour en savoir plus sur l'utilisation des clés de condition dans une politique, consultez la section Éléments de politique JSON IAM : Condition. La clé de condition `iam:PassedToService` peut être utilisée pour spécifier le principal de service du service auquel un rôle peut être transmis. Pour en savoir plus sur l'utilisation de la clé de `iam:PassedToService` condition dans une politique, consultez iam : PassedToService.

Exemple 1

Supposons que vous vouliez accorder à un utilisateur la possibilité de transférer l'un des ensembles de rôles approuvés au service Amazon EC2 lors du lancement d'une instance. Vous avez besoin de trois éléments :

- Une politique d'autorisations IAM attachée au rôle qui détermine les opérations pouvant être exécutées par le rôle. Limitez les autorisations aux seules actions nécessaires pour le rôle et aux seules ressources dont le rôle a besoin pour exécuter ces actions. Vous pouvez utiliser une politique d' AWS autorisations IAM gérée ou créée par le client.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [ "A list of the permissions the role is allowed to use" ],
        "Resource": [ "A list of the resources the role is allowed to access" ]
    }
}
```

- Une politique d'approbation pour le rôle, qui permet au service d'endosser le rôle. Par exemple, vous pourriez attacher la politique d'approbation suivante au rôle avec l'action `UpdateAssumeRolePolicy`. Cette politique d'approbation permet à Amazon EC2 d'utiliser le rôle et les autorisations attachées au rôle.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "TrustPolicyStatementThatAllowsEC2ServiceToAssumeTheAttachedRole",
        "Effect": "Allow",
        "Principal": { "Service": "ec2.amazonaws.com" },
       "Action": "sts:AssumeRole"
    }
}
```

- Une politique d'autorisations IAM attachée à l'utilisateur IAM, qui permet à l'utilisateur de transférer uniquement les rôles qui sont approuvés. Vous ajoutez généralement `iam:GetRole` à `iam:PassRole` pour que l'utilisateur puisse obtenir les détails du rôle à transférer. Dans cet exemple, l'utilisateur peut transférer uniquement les rôles qui existent dans le compte spécifié avec des noms commençant par l'interface `EC2-roles-for-XYZ-` :

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "iam:GetRole",
```

```
            "iam:PassRole"
        ],
        "Resource": "arn:aws:iam::account-id:role/EC2-roles-for-XYZ-*"
    }]
}
```

L'utilisateur peut maintenant démarrer une instance Amazon EC2 avec un rôle affecté. Les applications qui s'exécutent sur l'instance peuvent accéder à des informations d'identification temporaires pour le rôle via les métadonnées du profil d'instance. Les politiques d'autorisations attachées au rôle déterminent ce que peut faire l'instance.

Exemple 2

Amazon Relational Database Service (Amazon RDS) prend en charge une fonction appelée surveillance améliorée. Cette fonctionnalité permet à Amazon RDS de surveiller une instance de base de données grâce à un agent. Cela permet également à Amazon RDS d'enregistrer les métriques dans Amazon CloudWatch Logs. Pour activer cette fonctionnalité, vous devez créer un rôle de service pour accorder à Amazon RDS des autorisations pour surveiller et écrire des métriques dans vos journaux.

Pour créer un rôle pour la surveillance améliorée Amazon RDS

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Cliquez sur Rôles, puis sur Créer un rôle.
3. Choisissez le type AWS de rôle de service, puis pour Cas d'utilisation pour les autres Services AWS, choisissez le service RDS. Choisissez RDS - Enhanced Monitoring (RDS - Surveillance améliorée), puis Next (Suivant).
4. Choisissez la politique d'EnhancedMonitoringRoleautorisation d'Amazon RDS.
5. Choisissez Next (Suivant).
6. Pour Role name (Nom du rôle), saisissez un nom de rôle vous permettant d'identifier le but de ce rôle. Les noms de rôles doivent être uniques au sein de votre Compte AWS. Lorsqu'un nom de rôle est utilisé dans une politique ou dans le cadre d'un ARN, il est sensible à la casse. Lorsque le nom d'un rôle apparaît aux clients dans la console, par exemple lors du processus de connexion, il n'est pas sensible à la casse. Différentes entités peuvent référencer le rôle et il n'est donc pas possible de modifier son nom après sa création.
7. (Facultatif) Pour Description, saisissez une description pour le nouveau rôle.

8. (Facultatif) Ajoutez des métadonnées à l'utilisateur en associant les balises sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Balisage des ressources IAM.
9. Passez en revue les informations du rôle, puis choisissez Créer un rôle.

Le rôle obtient automatiquement une politique d'approbation qui accorde au service `monitoring.rds.amazonaws.com` les autorisations nécessaires pour endosser le rôle. Une fois cette opération effectuée, Amazon RDS peut exécuter toutes les actions que la politique `AmazonRDSEnhancedMonitoringRole` autorise.

L'utilisateur pour lequel vous souhaitez activer la surveillance améliorée a besoin d'une politique incluant une instruction lui permettant de lister les fonctions RDS et une autre lui prermettant de transférer la fonction comme suit. Utilisez votre numéro de compte et remplacez le nom du rôle par le nom fourni à l'étape 6.

```
    {
      "Sid": "PolicyStatementToAllowUserToListRoles",
      "Effect": "Allow",
      "Action": ["iam:ListRoles"],
      "Resource": "*"
    },
    {
        "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
        "Effect": "Allow",
        "Action": [ "iam:PassRole" ],
        "Resource": "arn:aws:iam::account-id:role/RDS-Monitoring-Role"
    }
```

Vous pouvez combiner cette instruction avec des instructions d'une autre politique ou la placer dans sa propre politique. Au lieu de spécifier que l'utilisateur peut transférer un rôle commençant par `RDS-`, vous pouvez remplacer le nom du rôle dans l'ARN d'une ressource par un caractère générique, comme suit.

```
        "Resource": "arn:aws:iam::account-id:role/RDS-*"
```

Actions **`iam:PassRole`** dans les journaux AWS CloudTrail

`PassRole`n'est pas un appel d'API. `PassRole`est une autorisation, ce qui signifie qu'aucun CloudTrail journal n'est généré pour IAM`PassRole`. Pour vérifier quels rôles sont transférés Services AWS à qui CloudTrail, vous devez consulter le CloudTrail journal qui a créé ou modifié la AWS ressource recevant le rôle. Par exemple, un rôle est transmis à une AWS Lambda fonction lors de sa création. le journal répertoriant l'action `CreateFunction` affichera l'enregistrement du rôle qui a été transmis à la fonction.

Changement de rôle (console)

Un rôle spécifie un ensemble d'autorisations que vous pouvez utiliser pour accéder aux ressources AWS dont vous avez besoin. À cet égard, il est semblable à un utilisateur IAM AWS Identity and Access Management. Lorsque vous vous connectez en tant qu'utilisateur, un ensemble spécifique d'autorisations vous est affecté. Toutefois, vous ne vous connectez pas au rôle mais, une fois connecté, vous pouvez endosser un rôle. Ceci annule temporairement vos autorisations utilisateur d'origine et les remplace par celles affectées au rôle. Le rôle peut être dans votre propre compte ou dans un autre Compte AWS. Pour plus d'informations sur les rôles, leurs avantages et la façon de les créer, consultez Rôles IAM et Création de rôles IAM.

> **Important**
>
> Les autorisations de votre utilisateur et des rôles que vous assumez ne peuvent pas être cumulées. Un seul ensemble d'autorisations peut être actif à la fois. Lorsque vous endossez un rôle, vous abandonnez temporairement vos autorisations utilisateur et travaillez avec celles qui sont affectées au rôle. Lorsque vous quittez le rôle, vos autorisations utilisateur sont automatiquement restaurées.

Lorsque vous changez de rôle dans le AWS Management Console, la console utilise toujours vos informations d'identification d'origine pour autoriser le changement. Ceci s'applique que vous vous connectiez en tant qu'utilisateur IAM, utilisateur de IAM Identity Center, rôle fédéré SAML ou rôle fédéré d'identité Web. Par exemple, si vous basculez sur RoleA, IAM utilise les informations d'identification d'utilisateur ou du rôle fédéré d'origine pour déterminer si vous êtes autorisé à endosser le rôle RoleA. Si vous passez ensuite à RoleB alors que vous utilisez RoleA, utilisez toujours vos informations d'identification d'utilisateur ou de rôle fédéré d'origine pour autoriser le changement AWS , et non les informations d'identification de RoleA.

Ce qu'il faut savoir sur le changement de rôles dans la console

Cette section fournit des informations supplémentaires sur l'utilisation de la console IAM pour endosser un rôle.

> **Remarques :**
>
> - Vous ne pouvez pas changer de rôle si vous vous connectez en tant que Utilisateur racine d'un compte AWS. Vous pouvez basculer d'un rôle à l'autre lorsque vous vous connectez en tant qu'utilisateur IAM, utilisateur de IAM Identity Center, rôle fédéré SAML ou rôle fédéré d'identité Web.
> - Vous ne pouvez pas changer de rôle dans le rôle AWS Management Console pour un rôle qui nécessite une ExternalIdvaleur. Vous ne pouvez basculer vers un tel rôle qu'en appelant l'API `AssumeRole` qui prend en charge le paramètre `ExternalId`.

- Si votre administrateur vous fournit un lien, cliquez dessus, puis passez à l'étape Step 5 de la procédure suivante. Le lien vous dirige vers la page web appropriée et renseigne les informations relatives à l'ID de compte (ou d'alias) et au nom de rôle.
- Vous pouvez créer manuellement le lien, puis passer à l'étape Step 5 de la procédure suivante. Pour créer votre lien, utilisez le format suivant :

  ```
  https://signin.aws.amazon.com/switchrole?
  account=account_id_number&roleName=role_name&displayName=text_to_display
  ```

  Où vous remplacez le texte suivant :

  - *`account_id_number`* : identifiant de compte à 12 chiffres fourni par votre administrateur. Votre administrateur peut également créer un alias de compte de manière à ce que l'URL contienne un nom au lieu d'un ID de compte. Pour plus d'informations, consultez Types d'utilisateurs dans le Guide de l'utilisateur Connexion à AWS .
  - *`role_name`* : nom du rôle que vous voulez endosser. Vous pouvez obtenir ce nom à partir de la fin de l'ARN du rôle. Par exemple, fournissez le nom de rôle `TestRole` à partir de l'ARN de rôle suivant : `arn:aws:iam::123456789012:role/TestRole`.
  - (Facultatif) *`text_to_display`* : texte que vous voulez afficher sur la barre de navigation à la place de votre nom d'utilisateur lorsque le rôle est actif.

- Vous pouvez changer manuellement de rôle à l'aide des informations fournies par votre administrateur en procédant comme suit.

Par défaut, lorsque vous changez de rôle, votre AWS Management Console session dure 1 heure. Par défaut, la durée des sessions utilisateur IAM est de 12 heures. Les utilisateurs IAM qui changent de rôle dans la console se voient accorder la durée de session maximale du rôle ou le temps restant dans la session de l'utilisateur IAM, selon la durée la plus courte. Par exemple, supposons qu'une durée de session maximale de 10 heures soit définie pour un rôle. Un utilisateur IAM s'est connecté à la console pendant 8 heures lorsqu'il décide d'endosser le rôle. Il reste 4 heures dans la session utilisateur IAM, de sorte que la durée autorisée de la session du rôle est de 4 heures. Le tableau suivant montre comment déterminer la durée de session d'un utilisateur IAM lorsqu'il change de rôle dans la console.

Durée de session de rôle d'un utilisateur IAM dans la console

Le temps restant de la session de l'utilisateur IAM est...	La durée de la session de rôle est…
Moins que la durée de session maximale d'un rôle	Temps restant dans la session de l'utilisateur IAM
Plus que la durée de session maximale du rôle	Valeur de durée de session maximale
Égale à la durée de session maximale du rôle	Valeur de durée de session maximale (approximative)

> **Note**
>
> Certaines consoles AWS de service peuvent renouveler automatiquement votre session de rôle lorsqu'elle expire sans que vous n'ayez à effectuer aucune action. Certaines peuvent vous inviter à recharger la page de votre navigateur pour réauthentifier votre session.

Pour résoudre les problèmes courants que vous pouvez rencontrer lorsque vous endossez un rôle, consultez Je ne parviens pas à endosser un rôle.

Pour passer à un rôle (console)

1. Connectez-vous en AWS Management Console tant qu'utilisateur IAM et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans la console IAM, sélectionnez votre nom d'utilisateur dans la barre de navigation, en haut à droite. Elle devrait afficher les informations suivantes : ***nom_utilisateur***@***ID_de_compte_ou_alias***.
3. Choisissez Changer de rôle. Si vous choisissez cette option pour la première fois, une page contenant plus d'informations s'affiche. Après l'avoir lue, choisissez Switch Role (Changer de rôle). Si vous désactivez les cookies de votre navigateur, cette page peut s'afficher de nouveau.
4. Dans la page Switch Role (Changer de rôle), entrez l'ID ou l'alias de compte, ainsi que le nom du rôle que vous a fourni l'administrateur.

   > **Note**
   >
   > Si l'administrateur a créé le rôle avec un chemin d'accès tel que `division_abc/subdivision_efg/roleToDoX`, vous devez entrer le chemin d'accès complet et le nom dans le champ Rôle. Si vous entrez uniquement le nom du rôle, ou si l'ensemble de `Path` et `RoleName` dépasse 64 caractères, le changement de rôle échoue. Cette restriction est imposée par les cookies du navigateur dans lesquels est stocké le nom du rôle. Dans ce cas, contactez votre administrateur et demandez-lui de réduire la taille du chemin d'accès et le nom du rôle.

5. (Facultatif) Choisissez un Nom d'affichage. Saisissez le texte que vous voulez afficher sur la barre de navigation à la place de votre nom utilisateur lorsque le rôle est actif. Un nom, basé sur les informations du compte et du rôle, est suggéré, mais vous pouvez le modifier à votre convenance. Il est également possible de sélectionner une couleur afin de mettre en évidence

le nom d'affichage. Le nom et la couleur peuvent vous aider à savoir quand le rôle est actif, ce qui modifie vos autorisations. Par exemple, pour un rôle qui vous donne accès à l'environnement de test, vous pouvez spécifier un Display name (Nom d'affichage) de **`Test`** et sélectionner la couleur verte pour Color. Pour le rôle qui vous donne accès à la production, vous pouvez spécifier un Display name (Nom d'affichage) de **`Production`** et sélectionner la couleur rouge pour Color.

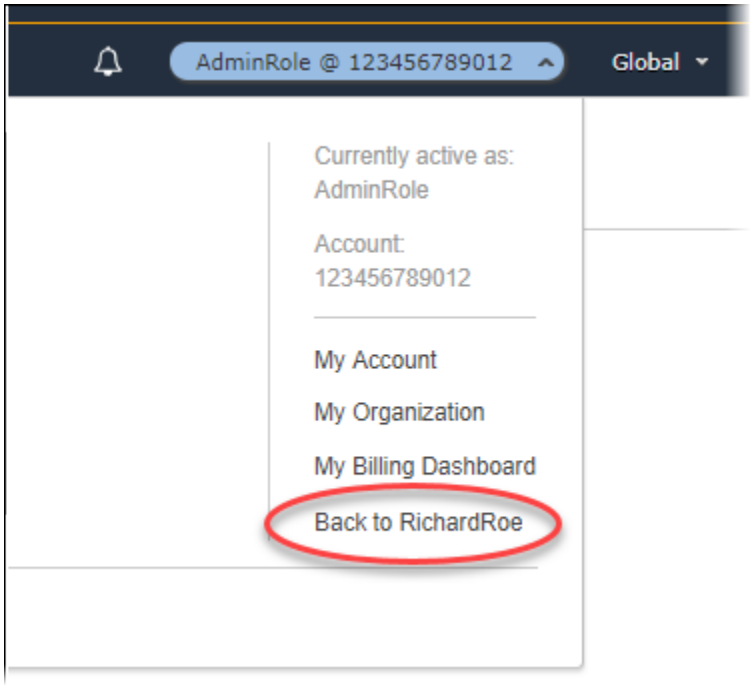
6. Choisissez Changer de rôle. Le nom d'affichage et la couleur remplacent votre nom utilisateur sur la barre de navigation et vous pouvez commencer à utiliser les autorisations que le rôle vous accorde.

> **Conseil**
>
> Les derniers rôles utilisés sont répertoriés dans le menu . La prochaine fois que vous devez endosser l'un de ces rôles, choisissez simplement le rôle souhaité. Il vous suffit de saisir manuellement les informations sur le compte et le rôle si le rôle n'est pas affiché dans le menu.

Pour cesser d'utiliser un rôle (console)

1. Dans la console IAM, choisissez le nom d'affichage du rôle sous Display Name (Nom d'affichage) en haut à droite de la barre de navigation. Il ressemble généralement à ceci : ***`nom_rôle`***@***`ID_compte_ou_alias`***.
2. Choisissez Back to ***`nom_utilisateur`*** (Revenir à nom_utilisateur). Le rôle et ses autorisations sont désactivés et les autorisations associées à votre utilisateur et vos groupes IAM sont automatiquement restaurées.

   Par exemple, supposons que vous êtes connecté au numéro de compte `123456789012` à l'aide du nom d'utilisateur `RichardRoe`. Une fois que vous avez utilisé le rôle `AdminRole`, vous souhaitez cesser d'utiliser ce rôle et revenir à vos autorisations initiales. Pour arrêter d'utiliser un rôle, choisissez AdminRole @ 123456789012, puis sélectionnez Retour à. RichardRoe



Assumer un rôle IAM (AWS CLI)

Un rôle spécifie un ensemble d'autorisations que vous pouvez utiliser pour accéder aux ressources AWS dont vous avez besoin. À cet égard, il est semblable à un utilisateur IAM AWS Identity and Access Management. Lorsque vous vous connectez en tant qu'utilisateur, un ensemble spécifique d'autorisations vous est affecté. Toutefois, vous ne vous connectez pas au rôle mais, une fois connecté en tant qu'utilisateur, vous pouvez passer à un rôle. Ceci annule temporairement vos autorisations utilisateur d'origine et les remplace par celles affectées au rôle. Le rôle peut être dans votre propre compte ou dans un autre Compte AWS. Pour plus d'informations sur les rôles, leurs avantages et la façon de les créer et de les configurer, consultez Rôles IAM et Création de rôles IAM. Pour connaître les différentes méthodes que vous pouvez utiliser pour endosser un rôle, consultez Utilisation de rôles IAM.

> **Important**
>
> Les autorisations de votre utilisateur IAM et des rôles que vous endossez ne peuvent pas être cumulées. Un seul ensemble d'autorisations peut être actif à la fois. Lorsque vous endossez un rôle, vous abandonnez temporairement les autorisations utilisateur et de rôle précédentes et utilisez celles qui sont affectées au rôle. Lorsque vous quittez le rôle, vos autorisations utilisateur sont automatiquement restaurées.

Vous pouvez utiliser un rôle pour exécuter une AWS CLI commande lorsque vous êtes connecté en tant qu'utilisateur IAM. Vous pouvez également utiliser un rôle pour exécuter une AWS CLI commande lorsque vous êtes connecté en tant qu'utilisateur authentifié de manière externe (SAML ou OIDC) qui utilise déjà un rôle. De plus, vous pouvez utiliser un rôle pour exécuter une commande de la AWS CLI dans une instance Amazon EC2 attachée à un rôle via son profil d'instance. Il n'est pas possible d'assumer un rôle si vous êtes connecté en tant qu' Utilisateur racine d'un compte AWS.

Chaînage de rôles : vous pouvez aussi utiliser le chaînage de rôles, qui utilise les autorisations d'un rôle pour accéder à un second rôle.

Par défaut, votre session de rôle dure une heure. Lorsque vous endossez ce rôle à l'aide des opérations de la CLI `assume-role*`, vous pouvez spécifier une valeur pour le paramètre `duration-seconds`. Cette valeur peut être comprise entre 900 secondes (15 minutes) et la valeur de durée de session maximale définie pour le rôle. Si vous changez de rôle dans la console, la durée de votre session est limitée à une heure. Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Affichage du paramètre de durée de session maximale pour un rôle.

Si vous utilisez la création de chaînes de rôles, la durée de votre session est limitée à une heure maximum. Si vous utilisez ensuite le paramètre `duration-seconds` pour fournir une valeur supérieure à une heure, l'opération échoue.

Exemple de scénario : passer à un rôle de production

Imaginez que vous êtes un utilisateur IAM pour travailler dans l'environnement de développement. Dans ce scénario, vous devez parfois travailler avec l'environnement de production au niveau de la ligne de commande avec l'AWS CLI. Vous disposez déjà d'un ensemble d'informations d'identification de clé d'accès. Il peut s'agir de la paire de clés d'accès attribuée à votre utilisateur IAM standard. Ou, si vous êtes connecté en tant qu'utilisateur fédéré, il peut s'agir de la paire de clés d'accès pour le rôle qui vous a été attribué initialement. Si vos autorisations actuelles vous permettent d'assumer un rôle IAM spécifique, vous pouvez identifier ce rôle dans un « profil » des fichiers de AWS CLI configuration. Cette commande est ensuite exécutée avec les autorisations du rôle IAM spécifié, pas l'identité d'origine. Notez que lorsque vous spécifiez ce profil dans une AWS CLI commande, vous utilisez le nouveau rôle. Dans ce cas, vous ne pouvez pas utiliser vos autorisations d'origine dans le compte de développement en même temps. Ceci est dû au fait qu'un seul ensemble d'autorisations peut être actif à la fois.

> **Note**
>
> Pour des raisons de sécurité, les administrateurs peuvent consulter AWS CloudTrail les journaux pour savoir qui a effectué une action dans AWS. Votre administrateur peut exiger que vous spécifiiez une identité source ou un nom de session de rôle lorsque vous endossez le rôle. Pour plus d'informations, consultez `sts:SourceIdentity` et `sts:RoleSessionName`.

Pour passer à un rôle de production (AWS CLI)

1. Si vous n'avez jamais utilisé le AWS CLI, vous devez d'abord configurer votre profil CLI par défaut. Ouvrez une invite de commande et configurez votre AWS CLI installation pour utiliser la clé d'accès de votre utilisateur IAM ou de votre rôle fédéré. Pour plus d'informations, veuillez consulter configuration de l'outil AWS Command Line Interface dans le guide de l'utilisateur de l'outil AWS Command Line Interface .

   Exécutez la commande aws configure comme suit :

   ```
   aws configure
   ```

   Lorsque vous y êtes invité, fournissez les informations suivantes :

   ```
   AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
   AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
   Default region name [None]: us-east-2
   Default output format [None]: json
   ```

2. Créez un profil pour le rôle dans le `.aws/config` fichier Unix ou Linux, ou le fichier `C:\Users\USERNAME\.aws\config` dans Windows. L'exemple suivant crée un profil appelé `prodaccess` qui passe au rôle *`ProductionAccessRole`* dans le compte `123456789012`. Vous obtenez l'ARN du rôle auprès de l'administrateur du compte ayant créé le rôle. Lorsque ce profil est invoqué, il AWS CLI utilise les informations d'identification du `source_profile` pour demander les informations d'identification du rôle. De ce fait, l'identité référencée en tant que `source_profile` doit disposer des autorisations `sts:AssumeRole` pour le rôle spécifié dans le `role_arn`.

   ```
   [profile prodaccess]
       role_arn = arn:aws:iam::123456789012:role/ProductionAccessRole
   ```

```
    source_profile = default
```

3. Après avoir créé le nouveau profil, toute AWS CLI commande spécifiant le paramètre est `--profile prodaccess` exécutée sous les autorisations associées au rôle IAM `ProductionAccessRole` plutôt que sous celles de l'utilisateur par défaut.

   ```
   aws iam list-users --profile prodaccess
   ```

   Cette commande fonctionne si les autorisations affectées au rôle `ProductionAccessRole` permettent de répertorier les utilisateurs dans le compte AWS actuel.

4. Pour revenir aux autorisations accordées par vos informations d'identification d'origine, exécutez les commandes sans le paramètre `--profile`. Vous AWS CLI recommencez à utiliser les informations d'identification de votre profil par défaut, que vous avez configuré dansStep 1.

Pour plus d'informations, consultez Endossement d'un rôle dans le Guide de l'utilisateur AWS Command Line Interface .

Exemple de scénario : permettre à un rôle de profil d'instance de passer à un rôle dans un autre compte

Imaginez que vous en utilisiez deux Comptes AWS et que vous souhaitiez autoriser une application exécutée sur une instance Amazon EC2 à exécuter des AWS CLIcommandes dans les deux comptes. Supposons que l'instance EC2 existe dans le compte `111111111111`. Cette instance inclut le rôle de profil d'instance `abcd` qui permet à l'appli d'effectuer les tâches Amazon S3 `my-bucket-1` en lecture seule sur le compartiment dans le même compte `111111111111`. Toutefois, l'application doit également être autorisée à endosser le rôle `efgh` entre comptes pour effectuer des tâches dans le compte `222222222222`. Pour ce faire, le rôle de profil d'instance EC2 `abcd` doit disposer des autorisations suivantes :

Politique d'autorisations de rôle ***abcd*** du compte 111111111111

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
```

```
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket-1/*",
                "arn:aws:s3:::my-bucket-1"
            ]
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

Supposons que le rôle `efgh` entre comptes permet aux tâches Amazon S3 en lecture seule sur le compartiment `my-bucket-2` dans le même compte `222222222222`. Pour ce faire, le rôle `efgh` entre comptes doit avoir la politique d'autorisations suivante :

Politique d'autorisations de rôle ***efgh*** du compte 222222222222

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
```

```
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket-2/*",
                "arn:aws:s3:::my-bucket-2"
            ]
        }
    ]
}
```

Le rôle `efgh` doit autoriser le rôle de profil d'instance `abcd` à l'endosser. Pour ce faire, le rôle `efgh` doit avoir la politique de confiance suivante :

Politique de confiance de rôle ***efgh*** du compte 222222222222

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

Pour exécuter ensuite AWS CLI les commandes dans le compte222222222222, vous devez mettre à jour le fichier de configuration de la CLI. Identifiez le rôle `efgh` en tant que « profil » et le rôle de profil d'instance EC2 `abcd` en tant que « source » d'informations d'identification dans le fichier de configuration d' AWS CLI . Ensuite, vos commandes de CLI sont exécutées avec les autorisations du rôle `efgh`, pas du rôle `abcd` d'origine.

> **Note**
>
> Pour des raisons de sécurité, vous pouvez l'utiliser AWS CloudTrail pour vérifier l'utilisation des rôles dans le compte. Pour différencier les sessions de rôle lorsqu'un rôle est utilisé par différents principaux dans les CloudTrail journaux, vous pouvez utiliser le nom de session du rôle. Lorsque le AWS CLI assume un rôle au nom d'un utilisateur tel que décrit dans cette rubrique, un nom de session de rôle est automatiquement créé sous le nom de`AWS-CLI-session-`*`nnnnnnnn`*. Ici, *`nnnnnnnn`* est un nombre entier qui représente le temps en Heure Unix (nombre de secondes écoulées depuis le 1er janvier 1970 à minuit UTC). Pour plus d'informations, consultez la section Référence aux CloudTrail événements dans le guide de AWS CloudTrail l'utilisateur.

Pour autoriser un rôle de profil d'instance EC2 afin de passer à un rôle entre comptes (AWS CLI)

1. Vous n'avez pas besoin de configurer un profil par défaut pour la CLI. Au lieu de cela, vous pouvez charger les informations d'identification à partir des métadonnées du profil d'instance EC2. Créez un profil pour le rôle dans le fichier `.aws/config`. L'exemple suivant crée un profil `instancecrossaccount` qui passe au rôle *`efgh`* dans le compte `222222222222`. Lorsque ce profil est appelé, l'interface AWS CLI utilise les informations d'identification des métadonnées du profil d'instance EC2 pour demander les informations d'identification du rôle. De ce fait, le rôle de profil d'instance EC2 doit disposer des autorisations `sts:AssumeRole` pour le rôle spécifié dans le `role_arn`.

   ```
   [profile instancecrossaccount]
   role_arn = arn:aws:iam::222222222222:role/efgh
   credential_source = Ec2InstanceMetadata
   ```

2. Après avoir créé le nouveau profil, toute AWS CLI commande spécifiant le paramètre `--profile instancecrossaccount` s'exécute sous les autorisations associées au `efgh` rôle dans le compte`222222222222`.

   ```
   aws s3 ls my-bucket-2 --profile instancecrossaccount
   ```

   Cette commande fonctionne si les autorisations affectées au rôle `efgh` autorisent à répertorier les utilisateurs dans l' Compte AWS actuel.

3. Pour revenir au profil d'instance EC2 d'origine les autorisations de compte `111111111111`, exécutez les commandes de CLI sans le paramètre `--profile`.

Pour plus d'informations, consultez Endossement d'un rôle dans le Guide de l'utilisateur AWS Command Line Interface .

Passer à un rôle IAM (Outils pour Windows PowerShell)

Un rôle spécifie un ensemble d'autorisations que vous pouvez utiliser pour accéder aux ressources AWS dont vous avez besoin. À cet égard, il est semblable à un utilisateur IAM AWS Identity and Access Management. Lorsque vous vous connectez en tant qu'utilisateur, un ensemble spécifique d'autorisations vous est affecté. Toutefois, vous ne vous connectez pas au rôle mais, une fois connecté, vous pouvez endosser un rôle. Ceci annule temporairement vos autorisations utilisateur d'origine et les remplace par celles affectées au rôle. Le rôle peut être dans votre propre compte ou dans un autre Compte AWS. Pour plus d'informations sur les rôles, leurs avantages et la façon de les créer et de les configurer, consultez Rôles IAM et Création de rôles IAM.

> **Important**
>
> Les autorisations de votre utilisateur IAM et des rôles que vous endossez ne peuvent pas être cumulées. Un seul ensemble d'autorisations peut être actif à la fois. Lorsque vous endossez un rôle, vous abandonnez temporairement vos autorisations utilisateur et travaillez avec celles qui sont affectées au rôle. Lorsque vous quittez le rôle, vos autorisations utilisateur sont automatiquement restaurées.

Cette section décrit comment changer de rôles lorsque vous utilisez la ligne de commande avec AWS Tools for Windows PowerShell.

Imaginez que vous possédez un compte dans l'environnement de développement et que vous deviez parfois travailler avec l'environnement de production en ligne de commande à l'aide des outils pour Windows PowerShell. Un ensemble d'informations d'identification de clé d'accès est déjà à votre disposition. Il peut s'agir de la paire de clés d'accès attribuée à votre utilisateur IAM standard. Ou, si vous êtes connecté en tant qu'utilisateur fédéré, il peut s'agir de la paire de clés d'accès pour le rôle qui vous a été attribué initialement. Ces informations d'identification vous permettent d'exécuter le cmdlet `Use-STSRole` qui transmet l'ARN d'un nouveau rôle en tant que paramètre. Cette commande renvoie es informations d'identification de sécurité temporaires du rôle demandé. Vous pouvez ensuite utiliser ces informations d'identification dans PowerShell les commandes suivantes avec les

autorisations du rôle pour accéder aux ressources en production. Bien que vous utilisiez ce rôle, vous ne pouvez pas utiliser vos autorisations d'utilisateur dans le compte Développement, car un seul ensemble d'autorisations est en vigueur à la fois.

> **Note**
>
> Pour des raisons de sécurité, les administrateurs peuvent consulter AWS CloudTrail les journaux pour savoir qui a effectué une action dans AWS. Votre administrateur peut exiger que vous spécifiiez une identité source ou un nom de session de rôle lorsque vous endossez le rôle. Pour plus d'informations, consultez `sts:SourceIdentity` et `sts:RoleSessionName`.

Notez que toutes les clés d'accès et tous les jetons sont des exemples uniquement et ne peuvent pas être utilisés comme indiqué. Remplacez-les par les valeurs correspondantes de votre environnement en direct.

Pour passer à un rôle (Outils pour Windows PowerShell)

1. Ouvrez une invite de PowerShell commande et configurez le profil par défaut pour utiliser la clé d'accès de votre utilisateur IAM actuel ou de votre rôle fédéré. Si vous avez déjà utilisé les outils pour Windows PowerShell, cela est probablement déjà fait. Notez que vous ne pouvez changer de rôle que si vous êtes connecté en tant qu'utilisateur IAM, et non en tant que Utilisateur racine d'un compte AWS.

   ```
   PS C:\> Set-AWSCredentials -AccessKey AKIAIOSFODNN7EXAMPLE -
   SecretKey wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY -StoreAs MyMainUserProfile
   PS C:\> Initialize-AWSDefaults -ProfileName MyMainUserProfile -Region us-east-2
   ```

   Pour plus d'informations, consultez la section Utilisation des AWS informations d'identification dans le guide de AWS Tools for Windows PowerShell l'utilisateur.

2. Pour récupérer les informations d'identification du nouveau rôle, exécutez la commande suivante pour passer au rôle *RoleName* dans le compte 123456789012. Vous obtenez l'ARN du rôle auprès de l'administrateur du compte ayant créé le rôle. La commande nécessite que vous fournissiez un nom de session également. Pour cela, n'importe quel texte fera l'affaire. La commande suivante demande les informations d'identification et capture l'objet de propriété `Credentials` dans l'objet des résultats renvoyés et le stocke dans la variable `$Creds`.

```
PS C:\> $Creds = (Use-STSRole -RoleArn "arn:aws:iam::123456789012:role/RoleName" -
RoleSessionName "MyRoleSessionName").Credentials
```

`$Creds` est un objet qui contient à présent les éléments `AccessKeyId`, `SecretAccessKey` et `SessionToken` dont vous avez besoin dans la procédure suivante. Les exemples de commandes suivants illustrent les valeurs habituelles.

```
PS C:\> $Creds.AccessKeyId
AKIAIOSFODNN7EXAMPLE

PS C:\> $Creds.SecretAccessKey
wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

PS C:\> $Creds.SessionToken
AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvSRyh0FW7jEXAMPLEW+vE/7s1HRp
XviG7b+qYf4nD00EXAMPLEmj4wxS04L/uZEXAMPLECihzFB5lTYLto9dyBgSDyEXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPy
Oj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UuysgsKdEXAMPLE1TVastU1A0SKFEXAMPLEiyw
C
s8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP+4eZScEXAMPLEsnf87eNhyDHq6ikBQ==

PS C:\> $Creds.Expiration
Thursday, June 18, 2018 2:28:31 PM
```

3. Pour utiliser ces informations d'identification dans n'importe quelle commande suivante, incluez-les dans le paramètre `-Credential`. Par exemple, la commande suivante utilise les informations d'identification du rôle et fonctionne uniquement si le rôle a l'autorisation `iam:ListRoles` et peut donc exécuter le cmdlet `Get-IAMRoles` :

```
PS C:\> get-iamroles -Credential $Creds
```

4. Pour revenir à vos informations d'identification d'origine, arrêtez simplement d'utiliser le `-Credentials $Creds` paramètre et PowerShell autorisez le retour aux informations d'identification stockées dans le profil par défaut.

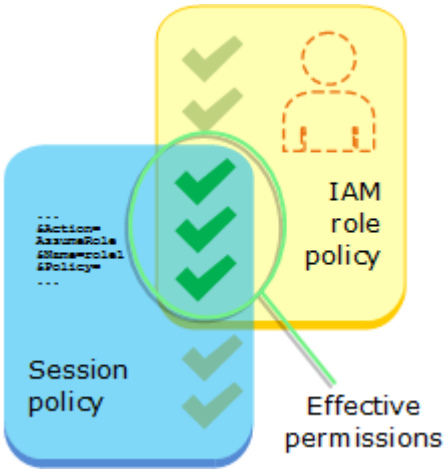
Passage à un rôle IAM (AWS API)

Un rôle spécifie un ensemble d'autorisations que vous pouvez utiliser pour accéder aux ressources AWS . À cet égard, il est semblable à un utilisateur IAM. Un principal (personne ou application) assume le rôle de recevoir des autorisations temporaires pour effectuer les tâches requises et interagir avec les AWS ressources. Le rôle peut être dans votre propre compte ou dans un autre Compte AWS. Pour plus d'informations sur les rôles, leurs avantages et la façon de les créer et de les configurer, consultez Rôles IAM et Création de rôles IAM. Pour connaître les différentes méthodes que vous pouvez utiliser pour endosser un rôle, consultez Utilisation de rôles IAM.

> **Important**
>
> Les autorisations de votre utilisateur IAM et des rôles que vous endossez ne peuvent pas être cumulées. Un seul ensemble d'autorisations peut être actif à la fois. Lorsque vous endossez un rôle, vous abandonnez temporairement les autorisations utilisateur et de rôle précédentes et utilisez celles qui sont affectées au rôle. Lorsque vous quittez le rôle, vos autorisations originales sont automatiquement restaurées.

Pour assumer un rôle, une application appelle l'opération AWS STS `AssumeRole`API et transmet l'ARN du rôle à utiliser. L'opération crée une nouvelle session avec des informations d'identification temporaires. Cette session possède les mêmes autorisations que les politiques basées sur une identité pour ce rôle.

Lorsque vous appelez `AssumeRole`, vous pouvez, si vous le souhaitez, transmettre des politiques de session en ligne ou gérées. Les politiques de session sont des politiques avancées que vous transmettez en tant que paramètre lorsque vous créez par programmation une session d'informations d'identification temporaires pour un rôle ou un utilisateur fédéré. Vous pouvez transmettre un seul document de politique de session en ligne JSON à l'aide du paramètre `Policy`. Vous pouvez utiliser le paramètre `PolicyArns` pour spécifier jusqu'à 10 politiques de session gérées. Les autorisations de la session obtenues sont une combinaison des stratégies basées sur l'identité de l'entité et des stratégies de session. Les politiques de session s'avèrent utiles lorsque vous devez fournir les informations d'identification temporaires du rôle à une autre personne. Cette dernière peut utiliser les informations d'identification temporaires du rôle dans les appels d'API AWS suivants pour accéder aux ressources du compte qui possède le rôle. Vous ne pouvez pas utiliser les politiques de session pour accorder plus d'autorisations que celles autorisées par la politique basée sur l'identité. Pour en savoir plus sur la manière de AWS déterminer les autorisations effectives d'un rôle, consultezLogique d'évaluation de politiques.



Vous pouvez appeler `AssumeRole` lorsque vous êtes connecté en tant qu'utilisateur IAM ou en tant qu'utilisateur authentifié en externe (SAML ou OIDC) utilisant déjà un rôle. Vous pouvez également utiliser la création de chaînes de rôles, qui utilise un rôle pour endosser un deuxième rôle. Il n'est pas possible d'assumer un rôle si vous êtes connecté en tant qu' Utilisateur racine d'un compte AWS.

Par défaut, votre session de rôle dure une heure. Lorsque vous assumez ce rôle à l'aide des opérations d' AWS STS `AssumeRole*`API, vous pouvez spécifier une valeur pour le `DurationSeconds` paramètre. Cette valeur peut être comprise entre 900 secondes (15 minutes) et la valeur de durée de session maximale définie pour le rôle. Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Affichage du paramètre de durée de session maximale pour un rôle.

Si vous utilisez la création de chaînes de rôles, votre session est limitée à une durée maximale d'une heure. Si vous utilisez ensuite le paramètre `DurationSeconds` pour fournir une valeur supérieure à une heure, l'opération échoue.

> **Note**
>
> Pour des raisons de sécurité, les administrateurs peuvent consulter AWS CloudTrail les journaux pour savoir qui a effectué une action dans AWS. Votre administrateur peut exiger que vous spécifiiez une identité source ou un nom de session de rôle lorsque vous endossez le rôle. Pour plus d'informations, consultez `sts:SourceIdentity` et `sts:RoleSessionName`.

Les exemples de code suivants montrent comment créer un utilisateur et assumer un rôle.

> **Warning**
>
> Afin d'éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

- Créer un utilisateur sans autorisation.
- Créer un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3 pour le compte.
- Ajouter une politique pour permettre à l'utilisateur d'assumer le rôle.
- Assumez le rôle et répertorier les compartiments S3 à l'aide d'informations d'identification temporaires, puis nettoyez les ressources.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;
```

```
public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
```

```
        var response = await _IAMService.AttachRolePolicyAsync(new
AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
{ GroupName = groupName });
        return response.Group;
    }


    /// <summary>
```

```
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }


    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
```

```
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
 { UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
 userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
 DeleteAccessKeyRequest
        {
```

```
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
```

```
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
 DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

```
   /// <summary>
   /// Delete an IAM user.
   /// </summary>
   /// <param name="userName">The username of the IAM user to delete.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DeleteUserAsync(string userName)
   {
       var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }


   /// <summary>
   /// Delete an IAM user policy.
   /// </summary>
   /// <param name="policyName">The name of the IAM policy to delete.</param>
   /// <param name="userName">The username of the IAM user.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
   {
       var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }


   /// <summary>
   /// Detach an IAM policy from an IAM role.
   /// </summary>
   /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
   /// <param name="roleName">The name of the IAM role.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
   {
       var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
```

```
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
```

```
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }
```

```
    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
 ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
```

```
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
_IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
```

```
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
 policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
```

```
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
```

```
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}
```

```
using Microsoft.Extensions.Configuration;

namespace IAMBasics;

public class IAMBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();


        IConfiguration configuration = new ConfigurationBuilder()
```

```
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();

        // First create a user. By default, the new user has
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
              $" \"AWS\": \"{userArn}\"" +
              "}," +
              "\"Action\": \"sts:AssumeRole\"" +
          "}]" +
        "}";

        // Permissions to list all buckets.
```

```
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);

        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;

        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");

        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");

        uiWrapper.PressEnter();
```

```
        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);

        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);

        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
```

```
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
 {bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
 complete.");
        Console.WriteLine("Please wait while we clean up the resources we
 created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();

        Console.WriteLine("All done cleaning up our resources. Thank you for your
 patience.");
    }
}
```

```
namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);
```

```
        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
{ BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
```

```
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
 { BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
 full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
```

```
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
```

```
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- Pour plus d’informations sur l’API, consultez les rubriques suivantes dans la référence de l’API AWS SDK for .NET .

- AttachRolePolitique
- CreateAccessClé
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessClé
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolitique
- DetachRolePolitique
- PutUserPolitique

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
```

```
##############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
  echo "This demo will create an IAM user, create an IAM role, and apply the role
 to the user."
  echo_repeat "*" 88
  echo

  echo -n "Enter a name for a new IAM user: "
  get_input
  user_name=$get_input_result

  local user_arn
  user_arn=$(iam_create_user -u "$user_name")

  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created demo IAM user named $user_name"
  else
    errecho "$user_arn"
    errecho "The user failed to create. This demo will exit."
    return 1
  fi

  local access_key_response
  access_key_response=$(iam_create_user_access_key -u "$user_name")
  # shellcheck disable=SC2181
  if [[ ${?} != 0 ]]; then
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
    return 1
  fi

  IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
  local key_name=${access_key_values[0]}
```

```
  local key_secret=${access_key_values[1]}

  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."

  local assume_role_policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi

  local policy_name
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]}"
```

```
 local policy_arn
 policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
 # shellcheck disable=SC2181
 if [[ ${?} == 0 ]]; then
   echo "Created  IAM policy named $policy_name"
 else
   errecho "The policy failed to create."
   clean_up "$user_name" "$key_name" "$iam_role_name"
   return 1
 fi

 if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
   echo "Attached policy $policy_arn to role $iam_role_name"
 else
   errecho "The policy failed to attach."
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
   return 1
 fi

 local assume_role_policy_document="{
              \"Version\": \"2012-10-17\",
              \"Statement\": [{
                  \"Effect\": \"Allow\",
                  \"Action\": \"sts:AssumeRole\",
                  \"Resource\": \"$role_arn\"}]}"

 local assume_role_policy_name
 assume_role_policy_name=$(generate_random_name "test-assume-role-")

 # shellcheck disable=SC2181
 local assume_role_policy_arn
 assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
 else
   errecho "The policy failed to create."
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
   return 1
 fi
```

```
 echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
 sleep 10
 echo_repeat "*" 88
 echo

 echo "Try to list buckets without the new user assuming the role."
 echo_repeat "*" 88
 echo

 # Set the environment variables for the created user.
 # bashsupport disable=BP2001
 export AWS_ACCESS_KEY_ID=$key_name
 # bashsupport disable=BP2001
 export AWS_SECRET_ACCESS_KEY=$key_secret

 local buckets
 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. This should not have
happened."
 else
   errecho "Because the role with permissions has not been assumed, listing
buckets failed."
 fi

 echo
 echo_repeat "*" 88
 echo "Now assume the role $iam_role_name and list the buckets."
 echo_repeat "*" 88
 echo

 local credentials

 credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
```

```
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  IFS=$'\t ' read -r -a credentials <<<"$credentials"

  export AWS_ACCESS_KEY_ID=${credentials[0]}
  export AWS_SECRET_ACCESS_KEY=${credentials[1]}
  # bashsupport disable=BP2001
  export AWS_SESSION_TOKEN=${credentials[2]}

  buckets=$(s3_list_buckets)

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. Listing buckets
succeeded because of "
    echo "the assumed role."
  else
    errecho "Failed to list buckets. This should not happen."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    export AWS_SESSION_TOKEN=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  local result=0
  export AWS_ACCESS_KEY_ID=""
  export AWS_SECRET_ACCESS_KEY=""

  echo
  echo_repeat "*" 88
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo
```

```
  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

Les fonctions IAM utilisées dans ce scénario.

```
##############################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
##############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
```

```
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
```

```
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

```
#############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
#############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
```

```
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
```

```
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-role \
    --role-name "$role_name" \
    --assume-role-policy-document "$policy_document" \
    --output text \
    --query Role.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
```

```
  echo "Creates an AWS Identity and Access Management (IAM) policy."
  echo "  -n policy_name   The name of the IAM policy."
  echo "  -p policy_json -- The policy document."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) policy_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_name" ]]; then
  errecho "ERROR: You must provide a policy name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-policy \
  --policy-name "$policy_name" \
  --policy-document "$policy_document" \
  --output text \
  --query Policy.Arn)

local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
```

```
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
```

```
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
```

```
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
```

```
  echo "Deletes an WS Identity and Access Management (IAM) policy"
  echo "  -n policy_arn -- The name of the IAM policy arn."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:h" option; do
  case "${option}" in
    n) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy arn with the -n parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Policy arn:  $policy_arn"
iecho ""

response=$(aws iam delete-policy \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
  return 1
fi

iecho "delete-policy response:$response"
```

```
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an WS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
```

```
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an WS Identity and Access Management (IAM) access key for the
specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
```

```
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
```

```
  echo ""
}

# Retrieve the calling parameters.
while getopts "u:h" option; do
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    User name:   $user_name"
iecho ""

# If the user does not exist, we don't want to try to delete it.
if (! iam_user_exists "$user_name"); then
  errecho "ERROR: A user with that name does not exist in the account."
  return 1
fi

response=$(aws iam delete-user \
  --user-name "$user_name")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-user operation failed.$response"
```

```
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la Référence des commandes AWS CLI .
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitique
  - DetachRolePolitique
  - PutUserPolitique

C++

SDK pour C++

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
namespace AwsDoc {
```

```
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;

    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
```

```
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
std::endl;
        }

        user = outcome.GetResult().GetUser();
    }

    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;

        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
```

```
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
    }
```

```
   // 3. Create an IAM policy.
   {
       Aws::IAM::Model::CreatePolicyRequest request;
       Aws::String uuid = Aws::Utils::UUID::RandomUUID();
       Aws::String policyName = "iam-demo-policy-" +
                                Aws::Utils::StringUtils::ToLower(uuid.c_str());
       request.SetPolicyName(policyName);

       // Build IAM policy document.
       Aws::Utils::Document jsonStatement;
       jsonStatement.WithString("Effect", "Allow");
       jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
       jsonStatement.WithString("Resource", "arn:aws:s3:::*");

       Aws::Utils::Document policyDocument;
       policyDocument.WithString("Version", "2012-10-17");

       Aws::Utils::Array<Aws::Utils::Document> statements(1);
       statements[0] = jsonStatement;
       policyDocument.WithArray("Statement", statements);

       std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                 << std::endl;

       // Set IAM policy document as JSON string.
       request.SetPolicyDocument(policyDocument.View().WriteCompact());

       Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
       if (!outcome.IsSuccess()) {
           std::cerr << "Error creating policy. " <<
                     outcome.GetError().GetMessage() << std::endl;

           DeleteCreatedEntities(client, role, user, policy);
           return false;
       }
       else {
           std::cout << "Successfully created a policy with name, " <<
policyName <<
                     "." << std::endl;
       }

       policy = outcome.GetResult().GetPolicy();
```

```
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);

        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;

                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
        }
```

```
        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());

        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
```

```
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }

            std::this_thread::sleep_for(std::chrono::seconds(1));
```

```
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}

bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
    bool result = true;
    if (policy.ArnHasBeenSet()) {
        // Detach the policy from the role.
        {
            Aws::IAM::Model::DetachRolePolicyRequest request;
            request.SetPolicyArn(policy.GetArn());
            request.SetRoleName(role.GetRoleName());

            Aws::IAM::Model::DetachRolePolicyOutcome outcome =
 client.DetachRolePolicy(
                    request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error Detaching policy from roles. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully detached the policy with arn "
                          << policy.GetArn()
                          << " from role " << role.GetRoleName() << "." <<
 std::endl;
            }
        }

        // Delete the policy.
        {
```

```
            Aws::IAM::Model::DeletePolicyRequest request;
            request.WithPolicyArn(policy.GetArn());

            Aws::IAM::Model::DeletePolicyOutcome outcome =
client.DeletePolicy(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error deleting policy. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully deleted the policy with arn "
                          << policy.GetArn() << std::endl;
            }
        }

    }

    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
```

```
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK for C++ .
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitique
  - DetachRolePolitique
  - PutUserPolitique

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Exécutez un scénario interactif à une invite de commande.

```
// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
// 1. Create a user who has no permissions.
// 2. Create a role that grants permission to list Amazon Simple Storage Service
//    (Amazon S3) buckets for the account.
// 3. Add a policy to let the user assume the role.
// 4. Try and fail to list buckets without permissions.
// 5. Assume the role and list S3 buckets using temporary credentials.
// 6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper actions.PolicyWrapper
 roleWrapper actions.RoleWrapper
 userWrapper actions.UserWrapper
 questioner demotools.IQuestioner
 helper IScenarioHelper
 isTestRun bool
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
  helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:   sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
  helper:         helper,
 }
}
```

```
// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run() {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
   log.Println(r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))

 user := scenario.CreateUser()
 accessKey := scenario.CreateAccessKey(user)
 role := scenario.CreateRoleAndPolicies(user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(accessKey)
 scenario.ListBucketsWithAssumedRole(noPermsConfig, role)
 scenario.Cleanup(user, role)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser() *types.User {
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
```

```
 user, err := scenario.userWrapper.GetUser(userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
  log.Printf("User %v already exists.\n", *user.UserName)
 }
 log.Println(strings.Repeat("-", 88))
 return user
}

// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(user *types.User)
 *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(*user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}

// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(user *types.User)
 *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err :=
 scenario.roleWrapper.CreateRole(scenario.helper.GetName(), *user.Arn)
```

```
 if err != nil {panic(err)}
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"}, "arn:aws:s3:::*")
 if err != nil {panic(err)}
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(*listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {panic(err)}
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
 err = scenario.userWrapper.CreateUserPolicy(*user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {panic(err)}
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(accessKey
 *types.AccessKey) *aws.Config {
  log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
  scenario.questioner.Ask("Press Enter when you're ready.")
  noPermsConfig, err := config.LoadDefaultConfig(context.TODO(),
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
 ))
  if err != nil {panic(err)}

  // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)
```

```
  s3Client := s3.NewFromConfig(noPermsConfig)
  _, err = s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
  if err != nil {
   // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
  }
  } else {
   log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
    "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
  return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
//
// 1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//   the user's access key credentials.
// 2. Gets temporary credentials by assuming the role that grants permission to
 list the
//    buckets.
// 3. Creates an Amazon S3 client from the temporary credentials.
// 4. Lists buckets for the account. Because the temporary credentials are
 generated by
//    assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(noPermsConfig
 *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
```

```
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(context.TODO(),
 &sts.AssumeRoleInput{
  RoleArn:           role.Arn,
  RoleSessionName:   aws.String("AssumeRoleExampleSession"),
  DurationSeconds:   aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(context.TODO(),
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
  ),
 )
 if err != nil {panic(err)}

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
 if err != nil {
  log.Println("Couldn't list buckets with assumed role credentials.")
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(user *types.User, role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
```

```
 ) {
   policies, err := scenario.roleWrapper.ListAttachedRolePolicies(*role.RoleName)
   if err != nil {panic(err)}
   for _, policy := range policies {
    err = scenario.roleWrapper.DetachRolePolicy(*role.RoleName,
 *policy.PolicyArn)
    if err != nil {panic(err)}
    err = scenario.policyWrapper.DeletePolicy(*policy.PolicyArn)
    if err != nil {panic(err)}
    log.Printf("Detached policy %v from role %v and deleted the policy.\n",
     *policy.PolicyName, *role.RoleName)
   }
   err = scenario.roleWrapper.DeleteRole(*role.RoleName)
   if err != nil {panic(err)}
   log.Printf("Deleted role %v.\n", *role.RoleName)

   userPols, err := scenario.userWrapper.ListUserPolicies(*user.UserName)
   if err != nil {panic(err)}
   for _, userPol := range userPols {
    err = scenario.userWrapper.DeleteUserPolicy(*user.UserName, userPol)
    if err != nil {panic(err)}
    log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
   }
   keys, err := scenario.userWrapper.ListAccessKeys(*user.UserName)
   if err != nil {panic(err)}
   for _, key := range keys {
    err = scenario.userWrapper.DeleteAccessKey(*user.UserName, *key.AccessKeyId)
    if err != nil {panic(err)}
    log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
 *user.UserName)
   }
   err = scenario.userWrapper.DeleteUser(*user.UserName)
   if err != nil {panic(err)}
   log.Printf("Deleted user %v.\n", *user.UserName)
   log.Println(strings.Repeat("-", 88))
 }

}
```

Définissez une structure qui encapsule les actions du compte.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy() (*types.PasswordPolicy,
 error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders() ([]types.SAMLProviderListEntry,
 error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

Définissez une structure qui encapsule les actions de la politique.

```
// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect string
 Action []string
 Principal map[string]string `json:",omitempty"`
 Resource *string `json:",omitempty"`
}



// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(maxPolicies int32) ([]types.Policy,
 error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(context.TODO(),
 &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
```

```
  policies = result.Policies
 }
 return policies, err
}



// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(policyName string, actions []string,
  resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(context.TODO(),
 &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

```
// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(policyArn string) (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(context.TODO(), &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(policyArn string) error {
 _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

Définissez une structure qui encapsule les actions du rôle.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(maxRoles int32) ([]types.Role, error) {
```

```
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(context.TODO(),
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(roleName string, trustedUserArn string)
 (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action: []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(context.TODO(),
 &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
```

```
 } else {
  role = result.Role
 }
 return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(roleName string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(context.TODO(),
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(serviceName string,
 description string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

```
// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(policyArn string, roleName string)
 error {
 _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),
 &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(roleName string)
 ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
```

```
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(roleName string, policyArn string)
 error {
 _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
 &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(roleName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteRole deletes a role. All attached policies must be detached before a
```

```
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteRole(context.TODO(), &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

Définissez une structure qui encapsule les actions de l'utilisateur.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(maxUsers int32) ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(context.TODO(), &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(userName string) (*types.User, error) {
```

```
 var user *types.User
 result, err := wrapper.IamClient.GetUser(context.TODO(), &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(context.TODO(),
 &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
```

```
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(userName string, policyName string,
 actions []string,
  roleArn string) error {
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(context.TODO(),
 &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(userName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
```

```
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(userName string, policyName string)
 error {
 _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
 &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(userName string) error {
 _, err := wrapper.IamClient.DeleteUser(context.TODO(), &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
```

```
func (wrapper UserWrapper) CreateAccessKeyPair(userName string)
 (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
 &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(userName string, keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
 &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
 &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
```

```
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Pour plus d'informations sur l'API consultez les rubriques suivantes dans la référence de l'API AWS SDK for Go .
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitique
  - DetachRolePolitique
  - PutUserPolitique

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de l'utilisateur IAM.

```
/*
```

```
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
 */

public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static String userArn;

    public static void main(String[] args) throws Exception {

        final String usage = """

                Usage:
                    <username> <policyName> <roleName> <roleSessionName>
 <bucketName>\s
```

```
                Where:
                    username - The name of the IAM user to create.\s
                    policyName - The name of the policy to create.\s
                    roleName - The name of the role to create.\s
                    roleSessionName - The name of the session required for the
assumeRole operation.\s
                    bucketName - The name of the Amazon S3 bucket from which
objects are read.\s
                """;

        if (args.length != 5) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        String policyName = args[1];
        String roleName = args[2];
        String roleSessionName = args[3];
        String bucketName = args[4];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the AWS IAM example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
```

```
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
```

```
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);
```

```
waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
           return response.policy().arn();

       } catch (IamException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
       return "";
   }

   public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
       try {
           ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                   .roleName(roleName)
                   .build();

           ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
           List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
           String polArn;
           for (AttachedPolicy policy : attachedPolicies) {
               polArn = policy.policyArn();
               if (polArn.compareTo(policyArn) == 0) {
                   System.out.println(roleName + " policy is already attached to
this role.");
                   return;
               }
           }

           AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                   .roleName(roleName)
                   .policyArn(policyArn)
                   .build();

           iam.attachRolePolicy(attachRequest);
           System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

       } catch (IamException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
    }

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();

        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();

            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
```

```
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void deleteRole(IamClient iam, String roleName, String polArn)
{

        try {
            // First the policy needs to be detached.
            DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                    .policyArn(polArn)
                    .roleName(roleName)
                    .build();

            iam.detachRolePolicy(rolePolicyRequest);

            // Delete the policy.
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(polArn)
                    .build();

            iam.deletePolicy(request);
            System.out.println("*** Successfully deleted " + polArn);

            // Delete the role.
            DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();
```

```
            iam.deleteRole(roleRequest);
            System.out.println("*** Successfully deleted " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteKey(IamClient iam, String username, String
 accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus d'informations sur l'API consultez les rubriques suivantes dans la référence de l'API AWS SDK for Java 2.x .
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitique
  - DetachRolePolitique
  - PutUserPolitique

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créer un utilisateur IAM et un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3. L'utilisateur n'a que le droit d'assumer le rôle. Après avoir assumé le rôle, utilisez des informations d'identification temporaires pour répertorier les compartiments pour le compte.

```
import {
  CreateUserCommand,
  GetUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
```

```
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import { ScenarioInput } from "@aws-doc-sdk-examples/lib/scenario/index.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "test_name";
const policyName = "test_policy";
const roleName = "test_role";

/**
 * Create a new IAM user. If the user already exists, give
 * the option to delete and re-create it.
 * @param {string} name
 */
export const createUser = async (name, confirmAll = false) => {
  try {
    const { User } = await iamClient.send(
      new GetUserCommand({ UserName: name }),
    );
    const input = new ScenarioInput(
      "deleteUser",
      "Do you want to delete and remake this user?",
      { type: "confirm" },
    );
    const deleteUser = await input.handle({}, { confirmAll });
    // If the user exists, and you want to delete it, delete the user
    // and then create it again.
    if (deleteUser) {
      await iamClient.send(new DeleteUserCommand({ UserName: User.UserName }));
      await iamClient.send(new CreateUserCommand({ UserName: name }));
    } else {
      console.warn(
        `${name} already exists. The scenario may not work as expected.`,
      );
```

```
      return User;
    }
  } catch (caught) {
    // If there is no user by that name, create one.
    if (caught instanceof Error && caught.name === "NoSuchEntityException") {
      const { User } = await iamClient.send(
        new CreateUserCommand({ UserName: name }),
      );
      return User;
    } else {
      throw caught;
    }
  }
};

export const main = async (confirmAll = false) => {
  // Create a user. The user has no permissions by default.
  const User = await createUser(userName, confirmAll);

  if (!User) {
    throw new Error("User not created");
  }

  // Create an access key. This key is used to authenticate the new user to
  // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
 (AWS STS).
  // It's not best practice to use access keys. For more information, see
 https://aws.amazon.com/iam/resources/best-practices/.
  const createAccessKeyResponse = await iamClient.send(
    new CreateAccessKeyCommand({ UserName: userName }),
  );

  if (
    !createAccessKeyResponse.AccessKey?.AccessKeyId ||
    !createAccessKeyResponse.AccessKey?.SecretAccessKey
  ) {
    throw new Error("Access key not created");
  }

  const {
    AccessKey: { AccessKeyId, SecretAccessKey },
  } = createAccessKeyResponse;

  let s3Client = new S3Client({
```

```
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
  // thrown while the user and access keys are still stabilizing.
  await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
    try {
      return await listBuckets(s3Client);
    } catch (err) {
      if (err instanceof Error && err.name === "InvalidAccessKeyId") {
        throw err;
      }
    }
  });

  // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
  // is thrown while the user and access keys are still stabilizing.
  const { Role } = await retry(
    {
      intervalInMs: 2000,
      maxRetries: 60,
    },
    () =>
      iamClient.send(
        new CreateRoleCommand({
          AssumeRolePolicyDocument: JSON.stringify({
            Version: "2012-10-17",
            Statement: [
              {
                Effect: "Allow",
                Principal: {
                  // Allow the previously created user to assume this role.
                  AWS: User.Arn,
                },
                Action: "sts:AssumeRole",
              },
            ],
          }),
          RoleName: roleName,
        }),
```

```
    ),
);

if (!Role) {
  throw new Error("Role not created");
}

// Create a policy that allows the user to list S3 buckets.
const { Policy: listBucketPolicy } = await iamClient.send(
  new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: ["s3:ListAllMyBuckets"],
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  }),
);

if (!listBucketPolicy) {
  throw new Error("Policy not created");
}

// Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
await iamClient.send(
  new AttachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
);

// Assume the role.
const stsClient = new STSClient({
  credentials: {
    accessKeyId: AccessKeyId,
    secretAccessKey: SecretAccessKey,
  },
});
```

```
// Retry the assume role operation until it succeeds.
const { Credentials } = await retry(
  { intervalInMs: 2000, maxRetries: 60 },
  () =>
    stsClient.send(
      new AssumeRoleCommand({
        RoleArn: Role.Arn,
        RoleSessionName: `iamBasicScenarioSession-${Math.floor(
          Math.random() * 1000000,
        )}`,
        DurationSeconds: 900,
      }),
    ),
);

if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
  throw new Error("Credentials not created");
}

s3Client = new S3Client({
  credentials: {
    accessKeyId: Credentials.AccessKeyId,
    secretAccessKey: Credentials.SecretAccessKey,
    sessionToken: Credentials.SessionToken,
  },
});

// List the S3 buckets again.
// Retry the list buckets operation until it succeeds. AccessDenied might
// be thrown while the role policy is still stabilizing.
await retry({ intervalInMs: 2000, maxRetries: 60 }, () =>
  listBuckets(s3Client),
);

// Clean up.
await iamClient.send(
  new DetachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
);

await iamClient.send(
  new DeletePolicyCommand({
```

```
      PolicyArn: listBucketPolicy.Arn,
    }),
  );

  await iamClient.send(
    new DeleteRoleCommand({
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeleteAccessKeyCommand({
      UserName: userName,
      AccessKeyId,
    }),
  );

  await iamClient.send(
    new DeleteUserCommand({
      UserName: userName,
    }),
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }

  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- Pour plus d'informations sur l'API consultez les rubriques suivantes dans la référence de l'API AWS SDK for JavaScript .
  - AttachRolePolitique

- CreateAccessClé
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessClé
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolitique
- DetachRolePolitique
- PutUserPolitique

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de l'utilisateur IAM.

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
 <bucketName>

    Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
 operation.
```

```
        fileLocation - The file location to the JSON required to create the role
 (see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
 read.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val userName = args[0]
    val policyName = args[1]
    val roleName = args[2]
    val roleSessionName = args[3]
    val fileLocation = args[4]
    val bucketName = args[5]

    createUser(userName)
    println("$userName was successfully created.")

    val polArn = createPolicy(policyName)
    println("The policy $polArn was successfully created.")

    val roleArn = createRole(roleName, fileLocation)
    println("$roleArn was successfully created.")
    attachRolePolicy(roleName, polArn)

    println("*** Wait for 1 MIN so the resource is available.")
    delay(60000)
    assumeGivenRole(roleArn, roleSessionName, bucketName)

    println("*** Getting ready to delete the AWS resources.")
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}

suspend fun createUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }
```

```
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {
    val policyDocumentValue: String =
        "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}"

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentValue
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}

suspend fun createRole(
    rolenameVal: String?,
    fileLocation: String?
): String? {
    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request =
        CreateRoleRequest {
            roleName = rolenameVal
            assumeRolePolicyDocument = jsonObject.toJSONString()
            description = "Created using the AWS SDK for Kotlin"
```

```
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(
    roleNameVal: String,
    policyArnVal: String
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String
```

```
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(
    roleArnVal: String?,
    roleSessionNameVal: String?,
    bucketName: String
) {
    val stsClient =
        StsClient {
            region = "us-east-1"
        }

    val roleRequest =
        AssumeRoleRequest {
            roleArn = roleArnVal
            roleSessionName = roleSessionNameVal
        }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials =
        StaticCredentialsProvider {
            accessKeyId = key
            secretAccessKey = secKey
            sessionToken = secToken
        }

    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 =
        S3Client {
```

```
            credentialsProvider = staticCredentials
            region = "us-east-1"
        }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects =
        ListObjectsRequest {
            bucket = bucketName
        }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}

suspend fun deleteRole(
    roleNameVal: String,
    polArn: String
) {
    val iam = IamClient { region = "AWS_GLOBAL" }

    // First the policy needs to be detached.
    val rolePolicyRequest =
        DetachRolePolicyRequest {
            policyArn = polArn
            roleName = roleNameVal
        }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
    val request =
        DeletePolicyRequest {
            policyArn = polArn
        }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")

    // Delete the role.
```

```
    val roleRequest =
        DeleteRoleRequest {
            roleName = roleNameVal
        }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient { region = "AWS_GLOBAL" }
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}

@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
    return jsonParser.parse(reader)
}
```

- Pour plus d'informations sur l'API consultez les rubriques suivantes dans la AWS Référence de l'API de SDK pour Kotlin.
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser

- DeleteUserPolitique
- DetachRolePolitique
- PutUserPolitique

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
namespace Iam\Basics;

require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
use Iam\IAMService;

echo("\n");
echo("---------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("---------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
               \"Version\": \"2012-10-17\",
               \"Statement\": [{
                   \"Effect\": \"Allow\",
                   \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
```

```
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
```

```
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}

$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK for PHP .
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy

- DeleteRole
- DeleteUser
- DeleteUserPolitique
- DetachRolePolitique
- PutUserPolitique

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créer un utilisateur IAM et un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3. L'utilisateur n'a que le droit d'assumer le rôle. Après avoir assumé le rôle, utilisez des informations d'identification temporaires pour répertorier les compartiments pour le compte.

```
import json
import sys
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
    print()
```

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, roles, and
 policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
```

```
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
            f"Couldn't create a policy and attach it to role {role.name}. Here's
why: "
            f"{error.response['Error']['Message']}"
        )
```

```
        raise

    try:
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
 "
            f"the role."
        )
    except ClientError as error:
        print(
            f"Couldn't create an inline policy for user {user.name}. Here's why:
 "
            f"{error.response['Error']['Message']}"
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
```

```
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
        "s3", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
 account.
    Uses the temporary credentials from the role to list the buckets that are
 owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the
 role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
```

```
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary
 credentials.
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise




def teardown(user, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
```

```
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        for user_pol in user.policies.all():
            user_pol.delete()
            print("Deleted inline user policy.")
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )


def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
        print("Thanks for watching!")


if __name__ == "__main__":
```

```
    usage_demo()
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Python (Boto3) API Reference.
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitique
  - DetachRolePolitique
  - PutUserPolitique

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créer un utilisateur IAM et un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3. L'utilisateur n'a que le droit d'assumer le rôle. Après avoir assumé le rôle, utilisez des informations d'identification temporaires pour répertorier les compartiments pour le compte.

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client
```

```
  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Waits for the specified number of seconds.
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts("Give AWS time to propagate resources...")
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Tried and failed to create demo user.")
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
```

```
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: "2012-10-17",
      Statement: [{
                   Effect: "Allow",
                   Principal: {'AWS': user.arn},
                   Action: "sts:AssumeRole"
                 }]
    }.to_json
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a role for the demo. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account,
and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
      Version: "2012-10-17",
      Statement: [{
                   Effect: "Allow",
                   Action: "s3:ListAllMyBuckets",
```

```
                  Resource: "arn:aws:s3:::*"
                }]
    }.to_json
    policy = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param user [Aws::IAM::User] The user that owns the policy.
  # @param role [Aws::IAM::Role] The role that can be assumed.
  # @return [Aws::IAM::UserPolicy] The newly created policy.
  def create_user_policy(policy_name, user, role)
    policy_document = {
      Version: "2012-10-17",
      Statement: [{
                  Effect: "Allow",
                  Action: "sts:AssumeRole",
                  Resource: role.arn
                }]
    }.to_json
    @iam_client.put_user_policy(
      user_name: user.user_name,
      policy_name: policy_name,
      policy_document: policy_document
    )
    puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
```

```
    @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an Amazon S3 resource with specified credentials. This is separated
into a
  # factory function so that it can be mocked for unit testing.
  #
  # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
  def create_s3_resource(credentials)
    Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
  end

  # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
  # Because the resource uses credentials with limited access, it may not be able
to
  # list the S3 buckets.
  #
  # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
  def list_buckets(s3_resource)
    count = 10
    s3_resource.buckets.each do |bucket|
      @logger.info "\t#{bucket.name}"
      count -= 1
      break if count.zero?
    end
  rescue Aws::Errors::ServiceError => e
    if e.code == "AccessDenied"
      puts("Attempt to list buckets with no permissions: AccessDenied.")
    else
      @logger.info("Couldn't list buckets for the account. Here's why: ")
      @logger.info("\t#{e.code}: #{e.message}")
      raise
    end
  end

  # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
  # This is separated into a factory function so that it can be mocked for unit
testing.
  #
```

```
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: "create-use-assume-role-scenario"
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end

 # Deletes a role. If the role has policies attached, they are detached and
 # deleted before the role is deleted.
 #
 # @param role_name [String] The name of the role to delete.
 def delete_role(role_name)
   @iam_client.list_attached_role_policies(role_name:
role_name).attached_policies.each do |policy|
     @iam_client.detach_role_policy(role_name: role_name, policy_arn:
policy.policy_arn)
     @iam_client.delete_policy(policy_arn: policy.policy_arn)
     @logger.info("Detached and deleted policy #{policy.policy_name}.")
   end
   @iam_client.delete_role({ role_name: role_name })
   @logger.info("Role deleted: #{role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
why:")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
```

```
  end

  # Deletes a user. If the user has inline policies or access keys, they are
 deleted
  # before the user is deleted.
  #
  # @param user [Aws::IAM::User] The user to delete.
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
  puts("-" * 88)
  puts("Welcome to the IAM create a user and assume a role demo!")
  puts("-" * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts("Try to list buckets with credentials for a user who has no permissions.")
  puts("Expect AccessDenied from this call.")
  scenario.list_buckets(
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key)))
  puts("Now, assume the role that grants permission.")
  temp_credentials = scenario.assume_role(
```

```
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key))
  puts("Here are your buckets:")
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts("Thanks for watching!")
  puts("-" * 88)
rescue Aws::Errors::ServiceError => e
  puts("Something went wrong with the demo.")
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- Pour plus d'informations sur l'API consultez les rubriques suivantes dans la référence de l'API AWS SDK for Ruby .
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitique
  - DetachRolePolitique
  - PutUserPolitique

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;

    if let Err(e) = run_iam_operations(
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
```

```
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;
```

```
    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
    );

    let attach_role_policy_result =
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
```

```
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
    match fail_client.list_buckets().send().await {
        Ok(e) => {
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
    let assumed_role = sts_client
        .assume_role()
        .role_arn(assume_role_role.arn())
        .role_session_name(&format!("{}{}", "iam_demo_assumerole_session_",
uuid))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
```

```
            .access_key_id(),
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
            println!("This should now run successfully.")
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
```

```
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Rust API reference.
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitique
  - DetachRolePolitique
  - PutUserPolitique

Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2

Les applications qui s'exécutent sur une instance Amazon EC2 doivent inclure des AWS informations d'identification dans les demandes d' AWS API. Vous pouvez demander à vos développeurs de stocker les AWS informations d'identification directement dans l'instance Amazon EC2 et d'autoriser les applications de cette instance à utiliser ces informations d'identification. Toutefois, dans ce cas,

ils doivent gérer les informations d'identification, les transmettre en toute sécurité à chaque instance, puis mettre à jour chaque instance Amazon EC2 lorsqu'il convient d'effectuer une mise à jour des informations d'identification. Ceci représente beaucoup de travail supplémentaire.

Au lieu de cela, il est recommandé d'utiliser un rôle IAM pour gérer les informations d'identification temporaires pour les applications qui s'exécutent sur une instance Amazon EC2. Lorsque vous utilisez un rôle, vous n'avez pas besoin de distribuer d'informations d'identification à long terme (telles que des informations de connexion ou des clés d'accès) à une instance Amazon EC2. Le rôle fournit plutôt des autorisations temporaires que les applications peuvent utiliser lorsqu'elles appellent d'autres AWS ressources. Lorsque vous lancez une instance Amazon EC2, vous spécifiez un rôle IAM à associer à celle-ci. Les applications qui s'exécutent sur l'instance peuvent ensuite utiliser les informations d'identification de sécurité temporaires fournies pour le rôle pour signer les demandes d'API.

L'utilisation de rôles pour l'octroi d'autorisations à des applications qui s'exécutent sur des instances Amazon EC2 requiert une configuration supplémentaire. Une application exécutée sur une instance Amazon EC2 est extraite du système AWS d'exploitation virtualisé. En raison de cette séparation supplémentaire, vous avez besoin d'une étape supplémentaire pour attribuer un AWS rôle et les autorisations associées à une instance Amazon EC2 et les mettre à la disposition de ses applications. Cette étape supplémentaire est la création d'un profil d'instance attaché à l'instance. Le profil d'instance contient le rôle et peut fournir les informations d'identification temporaires du rôle à une application qui s'exécute sur l'instance. Ces informations d'identification temporaires peuvent ensuite être utilisées dans les appels d'API de l'application pour accéder aux ressources et restreindre l'accès aux ressources spécifiées par le rôle uniquement.

> **Note**
>
> Chaque instance Amazon EC2 ne peut se voir attribuer qu'un seul rôle à la fois, et toutes les applications de l'instance partagent le même rôle et les mêmes autorisations. Lorsque vous exploitez Amazon ECS pour gérer vos instances Amazon EC2, vous pouvez attribuer des rôles aux tâches Amazon ECS qui peuvent être différenciés du rôle de l'instance Amazon EC2 sur laquelle elles s'exécutent. L'attribution d'un rôle à chaque tâche est conforme au principe de l'accès au moindre privilège et permet un contrôle plus précis des actions et des ressources.
> Pour plus d'informations, veuillez consulter la rubrique Using IAM roles with Amazon ECS tasks dans le Guide des bonnes pratiques pour Amazon Elastic Container Service.

Une telle utilisation des rôles présente plusieurs avantages. Dans la mesure où les informations d'identification des rôles sont temporaires et font automatiquement l'objet d'une mise à jour, vous n'avez pas à les gérer ni à vous soucier de risques de sécurité à long terme. Par ailleurs, si vous utilisez le même rôle pour plusieurs instances, toute modification apportée au rôle se propage automatiquement à toutes les instances.

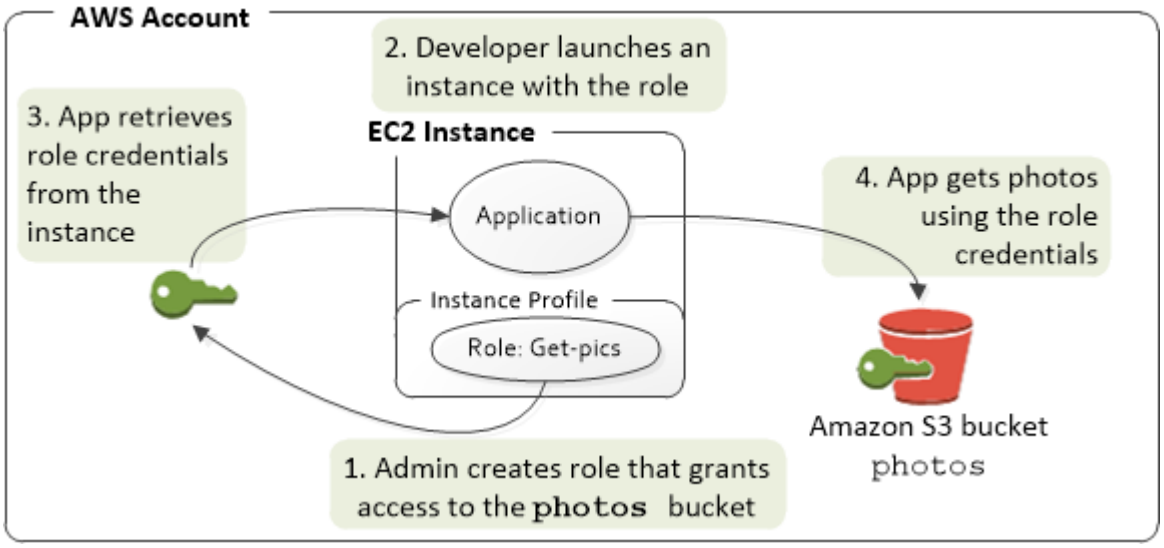
> **Note**
>
> Même si un rôle est généralement attribué à une instance Amazon EC2 lors du lancement, un rôle peut également être attaché à une instance Amazon EC2 en cours d'exécution. Pour savoir comment attacher un rôle à une instance en cours d'exécution, consultez Rôles IAM pour Amazon EC2.

Rubriques

- Comment fonctionnent les rôles pour les instances Amazon EC2 ?
- Autorisations requises pour l'utilisation de rôles avec Amazon EC2
- Comment puis-je commencer ?
- Informations connexes
- Utilisation de profils d'instance

Comment fonctionnent les rôles pour les instances Amazon EC2 ?

Dans l'illustration suivante, un développeur exécute une application sur une instance Amazon EC2 qui requiert l'accès à un compartiment S3 nommé `photos`. Un administrateur crée la fonction du service `Get-pics` et attache le rôle à l'instance Amazon EC2. Le rôle inclut une politique d'autorisations qui accorde un accès en lecture seule au compartiment S3 spécifié. Il inclut également une politique d'approbation qui permet à l'instance Amazon EC2 d'endosser le rôle et de récupérer les informations d'identification temporaires. Lorsque l'application s'exécute sur l'instance, elle peut accéder au compartiment photos en utilisant les informations d'identification temporaires du rôle. L'administrateur n'a pas besoin d'accorder au développeur l'autorisation d'accéder au compartiment photos et le développeur n'a à aucun moment besoin de partager ou gérer les informations d'identification.



1. L'administrateur utilise IAM pour créer le rôle **`Get-pics`**. Dans la politique d'approbation du rôle, l'administrateur spécifie que seules les instances Amazon EC2 peuvent endosser le rôle. Dans la politique d'autorisation du rôle, l'administrateur définit des autorisations en lecture seule pour le compartiment `photos`.
2. Un développeur lance une instance Amazon EC2 et lui affecte le rôle `Get-pics`.

   > **Note**
   >
   > Si vous utilisez la console IAM, le profil d'instance est géré de manière quasiment transparente pour vous. Toutefois, si vous utilisez l'API AWS CLI or pour créer et gérer le rôle et l'instance Amazon EC2, vous devez créer le profil d'instance et lui attribuer le rôle séparément. Ensuite, lorsque vous lancez l'instance, vous devez spécifier le nom du profil d'instance à la place du nom de rôle.

3. Lorsque l'application est en cours d'exécution, elle obtient des informations d'identification de sécurité temporaires à partir des métadonnées d'instance Amazon EC2, comme décrit dans Extraction des informations d'identification de sécurité à partir des métadonnées d'instance. Il s'agit d'informations d'identification de sécurité temporaires qui représentent le rôle et sont valides pendant une période limitée.

   Dans certains kits SDK AWS, le développeur peut utiliser un fournisseur qui gère de manière transparente les informations d'identification de sécurité temporaires. (La documentation des différents AWS SDK décrit les fonctionnalités prises en charge par ce SDK pour la gestion des informations d'identification.)

Sinon, l'application peut obtenir les informations d'identification temporaires directement à partir des métadonnées d'instance de l'instance Amazon EC2. Les informations d'identification et les valeurs associées se trouvent dans la catégorie `iam/security-credentials/`*`role-name`* (dans cet exemple, `iam/security-credentials/Get-pics`) des métadonnées. Si l'application extrait les informations d'identification des métadonnées de l'instance, elle peut les mettre en cache.

4. À l'aide des informations d'identification temporaires obtenues, l'application peut accéder au compartiment photos. Compte tenu de la politique attachée au rôle **`Get-pics`**, l'application dispose d'autorisations en lecture seule.

   Les informations d'identification de sécurité temporaires disponibles sur l'instance font automatiquement l'objet d'une mise à jour avant leur expiration, de manière à ce qu'un jeu d'identifiants valide soit toujours disponible. L'application doit simplement obtenir un nouvel ensemble d'informations d'identification à partir des métadonnées de l'instance avant l'expiration des informations d'identification actuelles. Il est possible d'utiliser le AWS SDK pour gérer les informations d'identification afin que l'application n'ait pas besoin d'inclure de logique supplémentaire pour actualiser les informations d'identification. Par exemple, instanciation de clients avec les fournisseurs d'informations d'identification de profil d'instance. En revanche, si l'application extrait les informations d'identification de sécurité temporaires des métadonnées d'instance, puis les met en cache, elle doit obtenir un ensemble d'informations d'identification actualisé toutes les heures, ou au moins 15 minutes avant l'expiration des informations d'identification actuelles. Le délai d'expiration est inclus dans les informations renvoyées dans la catégorie `iam/security-credentials/`*`role-name`*.

Autorisations requises pour l'utilisation de rôles avec Amazon EC2

Pour lancer une instance avec un rôle, le développeur doit avoir l'autorisation de lancer des instances Amazon EC2 et de transmettre des rôles IAM.

L'exemple de politique suivant permet aux utilisateurs d'utiliser le AWS Management Console pour lancer une instance avec un rôle. La politique inclut des caractères génériques (*) pour autoriser un utilisateur à transmettre n'importe quel rôle et à réaliser les actions Amazon EC2 énumérées. L'action `ListInstanceProfiles` permet aux utilisateurs d'afficher tous les rôles disponibles dans l'Compte AWS.

Example Exemple de politique qui autorise les utilisateurs à lancer une instance avec n'importe quel rôle à l'aide de la console Amazon EC2

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "IamPassRole",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Sid": "ListEc2AndListInstanceProfiles",
            "Effect": "Allow",
            "Action": [
                "iam:ListInstanceProfiles",
                "ec2:Describe*",
                "ec2:Search*",
                "ec2:Get*"
            ],
            "Resource": "*"
        }
    ]
}
```

Restreindre les rôles pouvant être transmis aux instances Amazon EC2 (en utilisant) PassRole

Vous pouvez utiliser l'autorisation `PassRole` pour spécifier les rôles qu'un utilisateur peut transmettre à une instance Amazon EC2 lors de son lancement. Cela évite que l'utilisateur exécute des applications ayant davantage d'autorisations qu'il n'en possède, autrement dit qu'il soit en mesure d'obtenir des droits élevés. Par exemple, supposons qu'une utilisatrice nommée Alice soit uniquement autorisée à lancer des instances Amazon EC2 et à utiliser des compartiments Amazon S3, mais que le rôle qu'elle transmet à une instance Amazon EC2 dispose d'autorisations permettant d'utiliser IAM et Amazon DynamoDB. Dans ce cas, il est possible qu'Alice soit en mesure de lancer l'instance, de s'y connecter, d'obtenir des informations d'identification de sécurité temporaires, puis d'effectuer plus d'actions IAM ou DynamoDB qu'elle n'est autorisée à le faire.

Pour limiter les rôles qu'un utilisateur peut transmettre à une instance Amazon EC2, vous pouvez créer une politique qui autorise l'action `PassRole`. Ensuite, vous attachez la politique à l'utilisateur (ou à un groupe IAM auquel il appartient) qui lancera les instances Amazon EC2. Dans l'élément `Resource` de la politique, vous répertoriez les rôles que l'utilisateur peut transmettre aux instances Amazon EC2. Lorsque l'utilisateur lance une instance et lui associe un rôle, Amazon EC2 vérifie si l'utilisateur est autorisé à transmettre ce rôle. Il est entendu que vous devez également vous assurer que le rôle que l'utilisateur est autorisé à transmettre n'inclut pas plus d'autorisations qu'il n'est censé en avoir.

> **Note**
>
> L'action d'API `PassRole` est différente de `RunInstances` ou `ListInstanceProfiles`. Il s'agit plutôt d'une autorisation qui AWS vérifie chaque fois qu'un ARN de rôle est transmis en tant que paramètre à une API (ou que la console le fait au nom de l'utilisateur). Elle permet à un administrateur de contrôler les rôles susceptibles d'être transmis et par quels utilisateurs. Dans le cas présent, elle veille à ce que l'utilisateur soit autorisé à attacher un rôle spécifique à une instance Amazon EC2.

Example Exemple de politique qui autorise un utilisateur à lancer une instance Amazon EC2 avec un rôle spécifique

Dans l'exemple suivant, la politique autorise les utilisateurs à lancer une instance avec un rôle à l'aide de l'API Amazon EC2. L'élément `Resource` spécifie l'Amazon Resource Name (ARN) d'un rôle. En spécifiant l'ARN, la politique accorde à l'utilisateur l'autorisation de transmettre uniquement le rôle `Get-pics`. Si l'utilisateur tente de spécifier un autre rôle lors du lancement d'une instance, l'action échoue. L'utilisateur est autorisé à exécuter n'importe quelle instance, qu'il s'agisse de transmettre un rôle ou non.

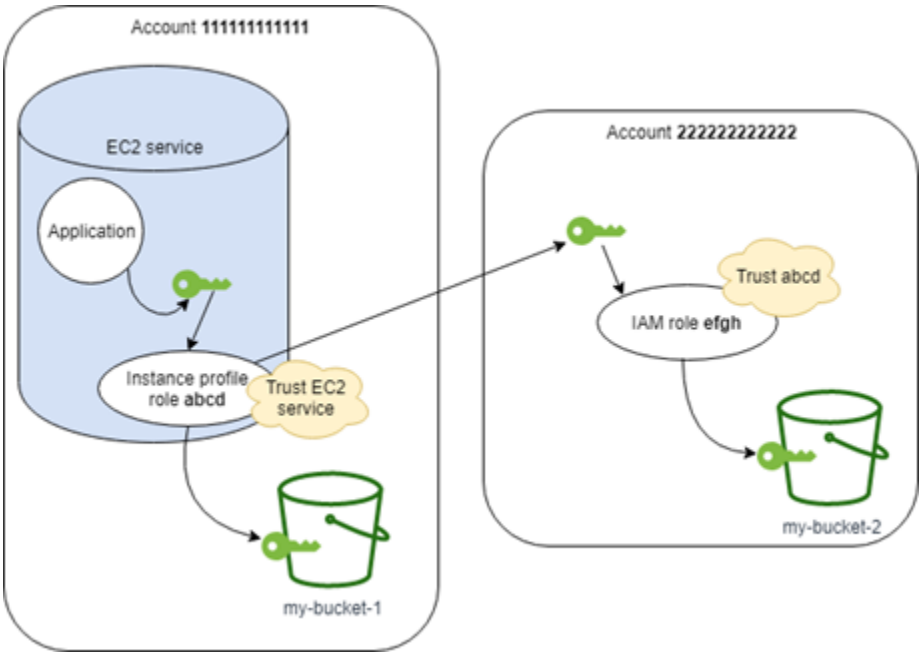
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
```

```
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::account-id:role/Get-pics"
    }
  ]
}
```

Permettre à un rôle de profil d'instance de passer à un rôle dans un autre compte

Vous pouvez permettre à une application qui s'exécute sur une instance Amazon EC2 d'exécuter des commandes dans un autre compte. Pour ce faire, vous devez autoriser le rôle d'instance Amazon EC2 dans le premier compte pour passer à un rôle dans le second compte.

Imaginez que vous en utilisiez deux Comptes AWS et que vous souhaitiez autoriser une application exécutée sur une instance Amazon EC2 à exécuter des AWS CLIcommandes dans les deux comptes. Supposons que l'instance Amazon EC2 existe dans le compte 111111111111. Cette instance inclut le rôle de profil d'instance `abcd` qui permet à l'appli d'effectuer les tâches Amazon S3 `my-bucket-1` en lecture seule sur le compartiment dans le même compte 111111111111. Toutefois, l'application doit également être autorisée à endosser le rôle `efgh` entre comptes pour accéder au compartiment `my-bucket-2` Amazon S3 dans le compte 222222222222.



Le rôle de profil d'instance Amazon EC2 `abcd` doit disposer des autorisations suivantes pour autoriser l'application à accéder au compartiment Amazon S3 `my-bucket-1` :

Politique d'autorisations de rôle ***abcd*** du compte 111111111111

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket-1/*",
                "arn:aws:s3:::my-bucket-1"
            ]
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

Le rôle `abcd` doit faire confiance au service Amazon EC2 pour endosser le rôle. Pour ce faire, le rôle `abcd` doit avoir la politique de confiance suivante :

Politique de confiance de rôle ***abcd*** du compte 111111111111

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "abcdTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"Service": "ec2.amazonaws.com"}
        }
    ]
}
```

Supposons que le rôle `efgh` entre comptes permet aux tâches Amazon S3 en lecture seule sur le compartiment `my-bucket-2` dans le même compte `222222222222`. Pour ce faire, le rôle `efgh` entre comptes doit avoir la politique d'autorisations suivante :

Politique d'autorisations de rôle ***`efgh`*** du compte 222222222222

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket-2/*",
                "arn:aws:s3:::my-bucket-2"
            ]
        }
    ]
}
```

Le rôle `efgh` doit autoriser le rôle de profil d'instance `abcd` à l'endosser. Pour ce faire, le rôle `efgh` doit avoir la politique de confiance suivante :

Politique de confiance de rôle ***efgh*** du compte 222222222222

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

Comment puis-je commencer ?

Pour comprendre le fonctionnement des rôles avec les instances Amazon EC2, vous devez utiliser la console IAM pour créer un rôle, lancer une instance Amazon EC2 utilisant ce rôle, puis examiner l'instance en cours d'exécution. Vous pouvez examiner les métadonnées de l'instance pour voir comment les informations d'identification temporaires du rôle sont mises à la disposition d'une instance. Il est également possible d'étudier la façon dont une application exécutée sur l'instance peut utiliser un rôle donné. Consultez les ressources suivantes pour en savoir plus.

-
- Procédures SDK. La documentation du AWS SDK inclut des procédures pas à pas qui montrent une application exécutée sur une instance Amazon EC2 qui utilise des informations d'identification temporaires pour les rôles afin de lire un compartiment Amazon S3. Chacune des procédures suivantes comporte des étapes similaires, avec un langage de programmation différent :
  - Configuration de rôles IAM pour Amazon EC2 avec le kit SDK for Java dans le Manuel du développeur AWS SDK for Java
  - Lancer une instance Amazon EC2 à l'aide du kit SDK pour .NET dans le Manuel du développeur AWS SDK for .NET
  - Création d'une instance Amazon EC2 avec le kit SDK for Ruby dans le Manuel du développeur AWS SDK for Ruby

Informations connexes

Pour en savoir plus sur la création de rôles pour des instances Amazon EC2, consultez les informations suivantes :

- Pour plus d'informations sur l'utilisation des rôles IAM avec les instances Amazon EC2, consultez le guide de l'utilisateur Amazon EC2.
- Pour créer un rôle, consultez Création de rôles IAM
- Pour plus d'informations sur l'utilisation d'informations d'identification de sécurité temporaires, consultez Informations d'identification de sécurité temporaires dans IAM.
- Si vous utilisez l'API ou l'interface de ligne de commande (CLI) IAM vous devez créer et gérer les profils d'instance IAM. Pour plus d'informations sur les profils d'instance, consultez Utilisation de profils d'instance.
- Pour plus d'informations sur les informations d'identification de sécurité temporaires pour les rôles dans les métadonnées de l'instance, consultez la section Extraction des informations d'identification de sécurité à partir des métadonnées de l'instance dans le guide de l'utilisateur Amazon EC2.

Utilisation de profils d'instance

Utilisez un profil d'instance pour transmettre un rôle IAM à une instance EC2. Pour plus d'informations, consultez la section Rôles IAM pour Amazon EC2 dans le guide de l'utilisateur Amazon EC2.

Gestion de profils d'instance (console)

Si vous utilisez le AWS Management Console pour créer un rôle pour Amazon EC2, la console crée automatiquement un profil d'instance et lui donne le même nom que le rôle. Lorsque vous utilisez ensuite la console Amazon EC2 pour lancer une instance avec un rôle IAM, vous pouvez sélectionner un rôle à associer à l'instance. Sur la console, la liste qui s'affiche est une liste de noms de profils d'instance. La console ne crée pas de profil d'instance pour un rôle qui n'est pas associé à Amazon EC2.

Vous pouvez utiliser le AWS Management Console pour supprimer des rôles IAM et des profils d'instance pour Amazon EC2 si le rôle et le profil d'instance portent le même nom. Pour en savoir plus sur la suppression de profils d'instance, veuillez consulter Suppression de rôles ou de profils d'instance.

Gestion des profils d'instance (AWS CLI ou AWS API)

Si vous gérez vos rôles à partir de l'API AWS CLI ou de l' AWS API, vous créez des rôles et des profils d'instance en tant qu'actions distinctes. Étant donné que les rôles et les profils d'instance peuvent porter des noms différents, vous devez connaître les noms de vos profils d'instance, ainsi que ceux des rôles qu'ils contiennent. De cette façon, vous pouvez choisir le profil d'instance correct lorsque vous lancez une instance EC2.

Vous pouvez attacher des balises à vos ressources IAM, y compris les profils d'instance, pour les identifier, les organiser et contrôler l'accès. Vous pouvez baliser les profils d'instance uniquement lorsque vous utilisez l' AWS API AWS CLI or.

> **Note**
>
> Un profil d'instance peut contenir un seul rôle IAM, mais un rôle peut être inclus dans plusieurs profils d'instance. Cette limite est d'un rôle par profil d'instance ne peut pas être augmentée. Vous pouvez supprimer le rôle existant, puis ajouter un autre rôle à un profil d'instance. Vous devez ensuite attendre que le changement apparaisse dans l'ensemble pour des AWS raisons de cohérence éventuelle. Pour forcer la modification, vous devez dissocier le profil d'instance, puis associer le profil d'instance, ou vous pouvez arrêter votre instance, puis la redémarrer.

Gestion de profils d'instance (AWS CLI)

Vous pouvez utiliser les AWS CLI commandes suivantes pour utiliser les profils d'instance d'un AWS compte.

- Création d'un profil d'instance : `aws iam create-instance-profile`
- Balisage d'un profil d'instance : `aws iam tag-instance-profile`
- Affichage de la liste des balises d'un profil d'instance : `aws iam list-instance-profile-tags`
- Suppression des balises d'un profil d'instance : `aws iam untag-instance-profile`
- Ajout d'un rôle à un profil d'instance : `aws iam add-role-to-instance-profile`
- Affichage d'une liste de profils d'instance : `aws iam list-instance-profiles`, `aws iam list-instance-profiles-for-role`
- Obtention d'informations sur un profil d'instance : `aws iam get-instance-profile`

- Suppression d'un rôle d'un profil d'instance : `aws iam remove-role-from-instance-profile`
- Suppression d'un profil d'instance : `aws iam delete-instance-profile`

Vous pouvez également attacher un rôle à une instance EC2 déjà en cours d'exécution à l'aide des commandes suivantes. Pour de plus amples informations, veuillez consulter la section Rôles IAM pour Amazon EC2.

- Attachement d'un profil d'instance avec un rôle à une instance EC2 arrêtée ou en cours d'exécution : `aws ec2 associate-iam-instance-profile`
- Obtention d'informations sur un profil d'instance attaché à une instance EC2 : `aws ec2 describe-iam-instance-profile-associations`
- Détachement d'un profil d'instance avec un rôle d'une instance EC2 arrêtée ou en cours d'exécution : `aws ec2 disassociate-iam-instance-profile`

Gestion de profils d'instance (API AWS)

Vous pouvez appeler les opérations d' AWS API suivantes pour travailler avec des profils d'instance dans un Compte AWS.

- Création d'un profil d'instance : `CreateInstanceProfile`
- Balisage d'un profil d'instance : `TagInstanceProfile`
- Affichage de la liste des balises d'un profil d'instance : `ListInstanceProfileTags`
- Suppression des balises d'un profil d'instance : `UntagInstanceProfile`
- Ajout d'un rôle à un profil d'instance : `AddRoleToInstanceProfile`
- Affichage d'une liste de profils d'instance : `ListInstanceProfiles`, `ListInstanceProfilesForRole`
- Obtention d'informations sur un profil d'instance : `GetInstanceProfile`
- Suppression d'un rôle d'un profil d'instance : `RemoveRoleFromInstanceProfile`
- Suppression d'un profil d'instance : `DeleteInstanceProfile`

Vous pouvez également attacher un rôle à une instance EC2 déjà en cours d'exécution en appelant les opérations suivantes. Pour de plus amples informations, veuillez consulter la section Rôles IAM pour Amazon EC2.

- Attachement d'un profil d'instance avec un rôle à une instance EC2 arrêtée ou en cours d'exécution : `AssociateIamInstanceProfile`
- Obtention d'informations sur un profil d'instance attaché à une instance EC2 : `DescribeIamInstanceProfileAssociations`
- Détachement d'un profil d'instance avec un rôle d'une instance EC2 arrêtée ou en cours d'exécution : `DisassociateIamInstanceProfile`

Révocation des informations d'identification de sécurité temporaires d'un rôle IAM

> **Warning**
>
> Si vous suivez les étapes indiquées sur cette page, tous les utilisateurs dont les sessions ont été créées en assumant le rôle se voient refuser l'accès à toutes les AWS actions et ressources. Cela peut entraîner le risque pour les utilisateurs de perdre leur travail non sauvegardé.

Lorsque vous autorisez les utilisateurs à accéder AWS Management Console à une session de longue durée (12 heures, par exemple), leurs informations d'identification temporaires n'expirent pas aussi rapidement. Si les utilisateurs exposent par inadvertance leurs informations d'identification à un tiers non autorisé, celui-ci dispose d'un accès pendant la durée de la session. Cependant, vous pouvez révoquer immédiatement toutes les autorisations aux informations d'identification du rôle émises avant un moment donné, si nécessaire. Toutes les informations d'identification temporaires pour ce rôle qui ont été émises avant le moment spécifié deviennent non valides. Cela force tous les utilisateurs à s'authentifier de nouveau et demande de nouvelles informations d'identification.

> **Note**
>
> Vous ne pouvez pas révoquer la session d'un rôle lié à un service.

Lorsque vous révoquez des autorisations pour un rôle à l'aide de la procédure décrite dans cette rubrique, AWS vous associez une nouvelle politique intégrée au rôle qui refuse toutes les autorisations pour toutes les actions. Il inclut une condition qui applique les restrictions uniquement si l'utilisateur a endossé le rôle avant le moment où vous avez révoqué les autorisations. Si l'utilisateur

endosse le rôle après la révocation des autorisations, la politique de rejet ne s'applique pas à cet utilisateur.

Pour plus d'informations sur le refus d'accès, veuillez consulter la rubrique Désactivation des autorisations affectées aux informations d'identification de sécurité temporaires.

> **Important**
>
> Cette politique de rejet s'applique à tous les utilisateurs du rôle spécifié, pas seulement à ceux dont les sessions de console durent plus longtemps.

Autorisations minimales pour révoquer les autorisations de session d'un rôle

Pour révoquer les autorisations de session d'un rôle avec succès, vous devez disposer de l'autorisation `PutRolePolicy` pour ce rôle. Cela vous permet d'attacher la politique en ligne `AWSRevokeOlderSessions` au rôle.

Révocation des autorisations de session

Vous pouvez révoquer les autorisations de session d'un rôle pour refuser toutes les autorisations à tout utilisateur qui a assumé le rôle.

> **Note**
>
> Vous ne pouvez pas modifier les rôles dans IAM qui ont été créés à partir des ensembles d'autorisations IAM Identity Center. Vous devez révoquer la session d'ensemble d'autorisations active pour un utilisateur dans IAM Identity Center. Pour plus d'informations, voir Révoquer les sessions de rôle IAM actives créées par des ensembles d'autorisations dans le guide de l'utilisateur d'IAM Identity Center.

Pour refuser immédiatement toutes autorisations à tout utilisateur actuel des informations d'identification du rôle

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Roles (Rôles), puis le nom (pas la case à cocher) du rôle dont vous voulez révoquer les autorisations.

3. Sur la page Récapitulatif du rôle sélectionné, choisissez l'onglet Révoquer les sessions.
4. Dans l'onglet Révoquer les sessions, choisissez Révoquer les sessions actives.
5. AWS vous demande de confirmer l'action. Cochez la case I acknowledge that I am revoking all active sessions for this role. (Je reconnais que je révoque toutes les sessions actives pour ce rôle.) et choisissez Revoke active sessions (Révoquer les sessions actives) dans la boîte de dialogue.

   IAM attache ensuite une politique nommée `AWSRevokeOlderSessions` au rôle. Une fois que vous avez choisi Révoquer les sessions actives, la politique refuse tout accès aux utilisateurs qui ont assumé le rôle dans le passé et environ 30 secondes dans le futur. Ce choix d'heure futur prend en compte le délai de propagation de la politique afin de traiter une nouvelle session acquise ou renouvelée avant que la politique mise à jour ne soit en vigueur dans une région donnée. Tout utilisateur qui assume le rôle plus de 30 secondes environ après avoir sélectionné Révoquer les sessions actives n'est pas concerné. Pour savoir pourquoi les modifications ne sont pas toujours visibles immédiatement, consultez la section Les modifications que j'apporte ne sont pas toujours visibles immédiatement.

> **Note**
>
> Si vous choisissez de révoquer à nouveau les sessions actives ultérieurement, la date et l'heure de la politique sont actualisées et toutes les autorisations sont à nouveau refusées à tout utilisateur ayant assumé le rôle avant la nouvelle heure spécifiée.

Les utilisateurs valides dont les sessions sont révoquées de cette manière doivent obtenir des informations d'identification temporaires pour qu'une nouvelle session continue de fonctionner. Le AWS CLI met en cache les informations d'identification jusqu'à leur expiration. Pour forcer la CLI à supprimer et actualiser les informations d'identification mises en cache qui ne sont plus valides, exécutez l'une des commandes suivantes :

Linux, macOS ou Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Révocation des autorisations de session avant une heure spécifiée

Vous pouvez également révoquer les autorisations de session à tout moment de votre choix à l'aide du SDK AWS CLI ou afin de spécifier une valeur pour la `lois : TokenIssue Heure` clé dans l'élément Condition d'une politique.

Cette politique refuse toutes les autorisations lorsque la valeur de `aws:TokenIssueTime` est antérieure aux date et heure spécifiées. La valeur de `aws:TokenIssueTime` correspond à l'exacte à laquelle les informations d'identification de sécurité temporaires ont été créées. La `aws:TokenIssueTime` valeur n'est présente que dans le contexte des AWS demandes signées avec des informations d'identification de sécurité temporaires, de sorte que l'instruction Deny de la politique n'affecte pas les demandes signées avec les informations d'identification à long terme de l'utilisateur IAM.

Cette politique peut également être attachée à un rôle. Dans ce cas, la politique affecte uniquement les informations d'identification de sécurité temporaires créées par le rôle avant la date et l'heure spécifiées.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {"aws:TokenIssueTime": "2014-05-07T23:47:00Z"}
    }
  }
}
```

Les utilisateurs valides dont les sessions sont révoquées de cette manière doivent obtenir des informations d'identification temporaires pour qu'une nouvelle session continue de fonctionner. Les informations d'identification sont mises en AWS CLI cache jusqu'à leur expiration. Pour forcer la CLI à supprimer et actualiser les informations d'identification mises en cache qui ne sont plus valides, exécutez l'une des commandes suivantes :

Linux, macOS ou Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Gestion des rôles IAM

Il peut arriver que vous deviez modifier ou supprimer les rôles que vous avez créés. Pour modifier un rôle, vous pouvez effectuer l'une des opérations suivantes :

- Modifiez toutes les politiques associées au rôle.
- Modifiez les personnes pouvant accéder au rôle.
- Modifiez les autorisations que le rôle accorde aux utilisateurs.
- Modifier le paramètre de durée maximale de session pour les rôles assumés à l'aide de AWS Management Console l'API AWS CLI ou

Vous pouvez également supprimer les rôles dont vous n'avez plus besoin. Vous pouvez gérer vos rôles depuis le AWS Management Console AWS CLI, le et l'API.

Rubriques

- Modification d'un rôle
- Suppression de rôles ou de profils d'instance

Modification d'un rôle

Vous pouvez utiliser l'API AWS Management Console AWS CLI, la ou l'API IAM pour apporter des modifications à un rôle.

Rubriques

- Afficher l'accès au rôle
- Générer une politique sur la base d'informations d'accès
- Modification d'un rôle (console)
- Modification d'un rôle (AWS CLI)
- Modifier un rôle (AWS API)

Afficher l'accès au rôle

Avant de modifier les autorisations d'un rôle, vous devez passer en revue ses activités récentes au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Générer une politique sur la base d'informations d'accès

Vous pouvez parfois octroyer plus d'autorisations à une entité IAM (utilisateur ou rôle) qu'elle n'en a besoin. Pour affiner les autorisations que vous octroyez, vous pouvez générer une politique IAM basée sur l'activité d'accès d'une entité. IAM Access Analyzer examine vos AWS CloudTrail journaux et génère un modèle de politique contenant les autorisations qui ont été utilisées par l'entité dans la plage de dates que vous avez spécifiée. Vous pouvez utiliser le modèle pour créer une politique gérée avec des autorisations affinées, puis l'attacher à l'entité IAM. Ainsi, vous accordez uniquement les autorisations dont l'utilisateur ou le rôle a besoin pour interagir avec les AWS ressources correspondant à votre cas d'utilisation spécifique. Pour en savoir plus, consultez Générer des politiques basées sur l'activité d'accès.

Modification d'un rôle (console)

Vous pouvez utiliser le AWS Management Console pour modifier un rôle. Pour modifier l'ensemble des balises sur un rôle, consultez Gestion des balises sur les rôles IAM (console).

Rubriques

- Modification d'une politique d'approbation de rôle (console)
- Modification d'une politique d'autorisations de rôle (console)
- Modification d'une description de rôle (console)
- Modification de la durée de session maximale d'un rôle (console)
- Modification d'une limite d'autorisations de rôle (console)

Modification d'une politique d'approbation de rôle (console)

Pour modifier la personne qui peut endosser un rôle, vous devez modifier la politique d'approbation du rôle. Vous ne pouvez pas modifier la politique d'approbation d'un rôle lié à un service.

Remarques

- Si un utilisateur est répertorié comme principal dans la politique d'approbation d'un rôle, mais ne peut pas endosser le rôle, vérifiez les limites d'autorisations de l'utilisateur. Si une limite d'autorisation est définie pour l'utilisateur, elle doit autoriser l'action `sts:AssumeRole`.
- Pour permettre aux utilisateurs d'assumer à nouveau le rôle actuel au cours d'une session de rôle, spécifiez l'ARN ou l' Compte AWS ARN comme principal dans la politique de confiance des rôles. Services AWS qui fournissent des ressources de calcul telles qu'Amazon EC2, Amazon ECS, Amazon EKS et Lambda fournissent des informations d'identification temporaires et mettent automatiquement à jour ces informations d'identification. Cela garantit que vous disposez toujours d'un ensemble d'informations d'identification valide. Pour ces services, il n'est pas nécessaire d'endosser à nouveau le rôle actuel pour obtenir des informations d'identification temporaires. Toutefois, si vous avez l'intention de transférer des balises de session ou une politique de session, vous devez endosser à nouveau le rôle actuel.

Pour modifier une politique d'approbation de rôle (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles).
3. Dans la liste des rôles de votre compte, choisissez le nom du rôle que vous souhaitez modifier.
4. Sélectionnez l'onglet Trust relationships (Relations d'approbation), puis Edit trust policy (Modifier la politique d'approbation).
5. Modifiez la politique d'approbation au besoin. Pour ajouter des principaux supplémentaires capables d'endosser le rôle, spécifiez-les dans l'élément `Principal`. Par exemple, l'extrait de politique suivant montre comment faire référence à deux éléments Comptes AWS dans l'`Principal`élément :

```
"Principal": {
  "AWS": [
    "arn:aws:iam::111122223333:root",
    "arn:aws:iam::444455556666:root"
```

```
  ]
},
```

Si vous spécifiez un principal dans un autre compte, l'ajout d'un compte à la politique d'approbation d'un rôle n'est qu'une partie de la procédure d'établissement de la relation d'approbation entre comptes. Par défaut, aucun utilisateur des comptes approuvés ne peut endosser le rôle. L'administrateur du compte nouvellement approuvé doit accorder aux utilisateurs l'autorisation d'endosser le rôle. Pour ce faire, l'administrateur doit créer ou modifier une politique attachée à l'utilisateur pour lui permettre d'accéder à l'action `sts:AssumeRole`. Pour plus d'informations, consultez la procédure suivante ou Octroi d'autorisations à un utilisateur pour endosser un rôle.

L'extrait de politique suivant montre comment référencer deux AWS services dans l'`Principal`élément :

```
"Principal": {
  "Service": [
    "opsworks.amazonaws.com",
    "ec2.amazonaws.com"
  ]
},
```

6. Après avoir modifié votre politique d'approbation, choisissez Mettre à jour la politique pour enregistrer vos modifications.

   Pour plus d'informations sur la syntaxe et la structure de la politique, consultez les sections Politiques et autorisations dans IAM et Références des éléments de politique JSON IAM.

Pour autoriser les utilisateurs d'un compte externe approuvé à utiliser le rôle (console)

Pour plus d'informations sur cette procédure, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle.

1. Connectez-vous à l'externe de confiance Compte AWS.
2. Décidez si vous devez attacher les autorisations à un utilisateur ou à un groupe. Dans le panneau de navigation de la console IAM, sélectionnez Users (Utilisateurs) ou User groups (Groupes d'utilisateurs) selon le cas.
3. Choisissez le nom de l'utilisateur ou du groupe auquel vous souhaitez accorder l'accès, puis sélectionnez l'onglet Autorisations.

4. Effectuez l'une des actions suivantes :

   - Pour modifier une politique gérée par un client, choisissez le nom de la politique, choisissez Modifier la politique, puis l'onglet JSON. Vous ne pouvez pas modifier une politique AWS gérée. AWS les politiques gérées apparaissent avec l' AWS icône (). Pour plus d'informations sur la différence entre les politiques AWS gérées et les politiques gérées par le client, consultezPolitiques gérées et politiques en ligne.
   - Pour modifier une politique en ligne, choisissez la flèche en regard du nom de la politique et choisissez Modifier la politique.
5. Dans l'éditeur de politiques, ajoutez un nouvel élément `Statement` spécifiant ce qui suit :

   ```
   {
     "Effect": "Allow",
     "Action": "sts:AssumeRole",
     "Resource": "arn:aws:iam::ACCOUNT-ID:role/ROLE-NAME"
   }
   ```

   Remplacez l'ARN dans l'instruction par celui du rôle que l'utilisateur peut endosser.
6. Suivez les invites à l'écran pour terminer de modifier la politique.

Modification d'une politique d'autorisations de rôle (console)

Pour modifier les autorisations accordées par le rôle, modifiez la stratégie des autorisations du rôle (ou les stratégies). Vous ne pouvez pas modifier la politique d'autorisations d'un rôle lié à un service dans IAM. Vous pouvez modifier la politique d'autorisations dans le service dépendant du rôle. Pour vérifier si un service prend en charge cette fonctionnalité, reportez-vous à la rubrique AWS services qui fonctionnent avec IAM et recherchez les services qui possèdent Yes (Oui)dans la colonne Rôles liés à un service. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Pour changer les autorisations autorisées par un rôle (console)

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles).
3. Choisissez le nom du rôle à modifier, puis choisissez l'onglet Autorisations.
4. Effectuez l'une des actions suivantes :

- Pour modifier une politique existante gérée par le client, choisissez le nom de la politique, puis choisissez Modifier la politique.

> **Note**
>
> Vous ne pouvez pas modifier une politique AWS gérée.
> AWS les politiques gérées apparaissent avec l' AWS icône
> ().
> Pour plus d'informations sur la différence entre les stratégies gérées AWS et les stratégies gérées par le client, consultez Politiques gérées et politiques en ligne.

- Pour attacher une stratégie gérée existante au rôle, choisissez Add permissions (Ajouter des autorisations) puis choisissez Attach policies (Attacher des politiques).
- Pour modifier une politique en ligne existante, développez la politique et choisissez Edit (Modifier).
- Pour intégrer une nouvelle politique en ligne, choisissez Add permissions (Ajouter des autorisations) puis choisissez Create inline policy (Création de stratégie en ligne).
- Pour supprimer une politique existante du rôle, cochez la case à côté du nom de la stratégie, puis choisissez Supprimer.

Modification d'une description de rôle (console)

Pour modifier la description du rôle, modifiez le texte de la description.

Pour modifier la description d'un rôle (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles).
3. Choisissez le nom du rôle à modifier.
4. Dans la section Summary (Récapitulatif), sélectionnez Edit (Modifier).
5. Saisissez une nouvelle description dans le champ et sélectionnez Save changes (Enregistrer les modifications).

Modification de la durée de session maximale d'un rôle (console)

Pour spécifier le paramètre de durée de session maximale pour les rôles assumés à l'aide de la console, de l' AWS CLI AWS API ou de l'API, modifiez la valeur du paramètre de durée maximale de session. La valeur de ce paramètre peut varier de 1 heure à 12 heures. Si vous ne spécifiez pas de valeur, la valeur par défaut de 1 heure maximum est appliquée. Ce paramètre ne limite pas les sessions endossées par des services AWS .

Pour modifier le paramètre de durée maximale de session pour les rôles assumés à l'aide de la console ou de AWS l'API (console) AWS CLI

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles).
3. Choisissez le nom du rôle à modifier.
4. Dans la section Summary (Récapitulatif), sélectionnez Edit (Modifier).
5. Pour Maximum session duration (Durée de session maximale), choisissez une valeur. Autrement, vous pouvez choisir Custom duration (Durée personnalisée) et saisir une valeur (en secondes).
6. Sélectionnez Enregistrer les modifications.

   Vos modifications ne prennent pas effet jusqu'à ce qu'une autre personne endosse ce rôle. Pour apprendre à révoquer des sessions existantes pour ce rôle, consultez Révocation des informations d'identification de sécurité temporaires d'un rôle IAM.

Dans le AWS Management Console, les sessions utilisateur IAM durent 12 heures par défaut. Les utilisateurs IAM qui changent de rôle dans la console se voient accorder la durée de session maximale du rôle ou le temps restant dans la session de l'utilisateur IAM, selon la durée la plus courte.

Toute personne assumant le rôle depuis l' AWS API AWS CLI or peut demander une session plus longue, jusqu'à ce maximum. Le paramètre `MaxSessionDuration` détermine la durée maximale de la session de rôle qui peut être demandée.

- Pour spécifier une durée de session à l'aide du paramètre, AWS CLI utilisez le `duration-seconds` paramètre. Pour en savoir plus, veuillez consulter la section Assumer un rôle IAM (AWS CLI).

- Pour spécifier une durée de session à l'aide de l' AWS API, utilisez le `DurationSeconds` paramètre. Pour en savoir plus, veuillez consulter la section Passage à un rôle IAM (AWS API).

Modification d'une limite d'autorisations de rôle (console)

Pour modifier le nombre maximum d'autorisations permises pour un rôle, modifiez la limite d'autorisations du rôle

Pour modifier la politique utilisée afin de définir la limite d'autorisations pour un rôle

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Roles (Rôles).
3. Sélectionnez le nom du rôle présentant la limite d'autorisations que vous souhaitez modifier.
4. Choisissez l'onglet Permissions (Autorisations). Si nécessaire, ouvrez la section Permissions boundary (Limite d'autorisations), puis choisissez Change boundary (Modifier une limite).
5. Sélectionnez la politique que vous souhaitez utiliser pour la limite d'autorisations.
6. Choisissez Change boundary (Modifier une limite).

   Vos modifications ne prennent pas effet jusqu'à ce qu'une autre personne endosse ce rôle.

Modification d'un rôle (AWS CLI)

Vous pouvez utiliser le AWS Command Line Interface pour modifier un rôle. Pour modifier l'ensemble des balises sur un rôle, consultez Gestion des balises sur les rôles IAM (AWS CLI ou AWS API).

Rubriques

- Modification d'une politique d'approbation de rôle (AWS CLI)
- Modification d'une politique d'autorisation de rôle (AWS CLI)
- Modification d'une description de rôle (AWS CLI)
- Modification de la durée de session maximale d'un rôle (AWS CLI)
- Modification d'une limite d'autorisations de rôle (AWS CLI)

Modification d'une politique d'approbation de rôle (AWS CLI)

Pour modifier la personne qui peut endosser un rôle, vous devez modifier la politique d'approbation du rôle. Vous ne pouvez pas modifier la politique d'approbation d'un rôle lié à un service.

> **Remarques**
>
> - Si un utilisateur est répertorié comme principal dans la politique d'approbation d'un rôle, mais ne peut pas endosser le rôle, vérifiez les limites d'autorisations de l'utilisateur. Si une limite d'autorisation est définie pour l'utilisateur, elle doit autoriser l'action `sts:AssumeRole`.
> - Pour permettre aux utilisateurs d'assumer à nouveau le rôle actuel au cours d'une session de rôle, spécifiez l'ARN ou l' Compte AWS ARN comme principal dans la politique de confiance des rôles. Services AWS qui fournissent des ressources de calcul telles qu'Amazon EC2, Amazon ECS, Amazon EKS et Lambda fournissent des informations d'identification temporaires et mettent automatiquement à jour ces informations d'identification. Cela garantit que vous disposez toujours d'un ensemble d'informations d'identification valide. Pour ces services, il n'est pas nécessaire d'endosser à nouveau le rôle actuel pour obtenir des informations d'identification temporaires. Toutefois, si vous avez l'intention de transférer des balises de session ou une politique de session, vous devez endosser à nouveau le rôle actuel. Pour savoir comment modifier une politique d'approbation des rôles afin d'ajouter l'ARN ou Compte AWS l'ARN du rôle principal, consultezModification d'une politique d'approbation de rôle (console).

Pour modifier une politique d'approbation de rôle (AWS CLI)

1. (Facultatif) Si vous ne connaissez pas le nom du rôle que vous souhaitez modifier, exécutez la commande suivante pour répertorier les rôles de votre compte :
   - aws iam list-roles
2. (Facultatif) Pour afficher la politique d'approbation actuelle d'un rôle, exécutez la commande suivante :
   - aws iam get-role
3. Pour modifier les principaux approuvés ayant accès au rôle, créez un fichier texte avec la politique d'approbation mise à jour. Vous pouvez utiliser l'éditeur de texte pour créer la politique.

Par exemple, la politique de confiance suivante montre comment faire référence à deux Comptes AWS dans l'`Principal`élément. Cela permet aux utilisateurs de deux Comptes AWS distincts d'endosser ce rôle.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {"AWS": [
            "arn:aws:iam::111122223333:root",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "sts:AssumeRole"
    }
}
```

Si vous spécifiez un principal dans un autre compte, l'ajout d'un compte à la politique d'approbation d'un rôle n'est qu'une partie de la procédure d'établissement de la relation d'approbation entre comptes. Par défaut, aucun utilisateur des comptes approuvés ne peut endosser le rôle. L'administrateur du compte nouvellement approuvé doit accorder aux utilisateurs l'autorisation d'endosser le rôle. Pour ce faire, l'administrateur doit créer ou modifier une politique attachée à l'utilisateur pour lui permettre d'accéder à l'action `sts:AssumeRole`. Pour plus d'informations, consultez la procédure suivante ou Octroi d'autorisations à un utilisateur pour endosser un rôle.

4. Pour utiliser le fichier créé afin de mettre à jour la politique de confiance, exécutez la commande suivante :

   - était un objectif update-assume-role-policy

Pour autoriser les utilisateurs d'un compte externe approuvé à utiliser le rôle (AWS CLI)

Pour plus d'informations sur cette procédure, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle.

1. Créez un fichier JSON contenant une politique d'autorisations qui accorde des autorisations permettant d'endosser le rôle. Par exemple, la politique suivante contient les autorisations nécessaires minimales :

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::ACCOUNT-ID-THAT-CONTAINS-ROLE:role/ROLE-NAME"
  }
}
```

Remplacez l'ARN dans l'instruction par celui du rôle que l'utilisateur peut endosser.

2. Exécutez la commande suivante pour charger le fichier JSON contenant la politique de confiance dans IAM :

   - aws iam create-policy

   Le résultat de cette commande inclut l'ARN de la politique. Notez cet ARN, car vous aurez besoin de l'utiliser ultérieurement.

3. Décidez à quel utilisateur ou groupe attacher la politique. Si vous ne connaissez pas le nom de l'utilisateur ou du groupe concerné, utilisez les commandes suivantes pour répertorier les utilisateurs ou les groupes de votre compte :

   - aws iam list-users
   - aws iam list-groups

4. Utilisez l'une des commandes suivantes pour attacher la politique créée dans l'étape précédente à l'utilisateur ou au groupe :

   - était un objectif attach-user-policy
   - était un objectif attach-group-policy

Modification d'une politique d'autorisation de rôle (AWS CLI)

Pour modifier les autorisations accordées par le rôle, modifiez la stratégie des autorisations du rôle (ou les stratégies). Vous ne pouvez pas modifier la politique d'autorisations d'un rôle lié à un service dans IAM. Vous pouvez modifier la politique d'autorisations dans le service dépendant du rôle. Pour vérifier si un service prend en charge cette fonctionnalité, reportez-vous à la rubrique AWS services qui fonctionnent avec IAM et recherchez les services qui possèdent Yes (Oui)dans la colonne Rôles

liés à un service. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Pour modifier les autorisations autorisées par un rôle (AWS CLI)

1. (Facultatif) Pour afficher les autorisations actuelles associées à un rôle, exécutez les commandes suivantes :
    1. aws vise à list-role-policies répertorier les politiques en ligne
    2. aws iam va list-attached-role-policies répertorier les politiques gérées
2. La commande permettant de mettre à jour les autorisations du rôle varie selon que vous mettez à jour une politique gérée ou une politique en ligne.

    Pour mettre à jour une politique gérée, exécutez la commande suivante pour créer une nouvelle version de la politique gérée :

    - était un objectif create-policy-version

    Pour mettre à jour une politique en ligne, exécutez la commande suivante :

    - était un objectif put-role-policy

Modification d'une description de rôle (AWS CLI)

Pour modifier la description du rôle, modifiez le texte de la description.

Pour modifier la description d'un rôle (AWS CLI)

1. (Facultatif) Pour afficher la description actuelle d'un rôle, exécutez la commande suivante :
    - aws iam get-role
2. Pour mettre à jour la description d'un rôle, exécutez la commande suivante avec le paramètre de description :
    - aws iam update-role

Modification de la durée de session maximale d'un rôle (AWS CLI)

Pour spécifier le paramètre de durée de session maximale pour les rôles qui sont endossés à l'aide de l’interface AWS CLI ou de l'API, modifiez la valeur du paramètre de durée de session maximale. La valeur de ce paramètre peut varier de 1 heure à 12 heures. Si vous ne spécifiez pas de valeur, la valeur par défaut de 1 heure maximum est appliquée. Ce paramètre ne limite pas les sessions assumées par AWS les services.

> **Note**
>
> Toute personne assumant le rôle depuis l'API AWS CLI or peut utiliser le paramètre `duration-seconds` CLI ou le paramètre `DurationSeconds` API pour demander une session plus longue. Le paramètre `MaxSessionDuration` détermine la durée maximale de la session de rôle qui peut être demandée à l'aide du paramètre `DurationSeconds`. Si les utilisateurs ne spécifient pas de valeur pour le paramètre `DurationSeconds`, leurs informations d'identification de sécurité sont valides pendant une heure.

Pour modifier le paramètre de durée de session maximale pour les rôles qui sont endossés à l'aide de l’interface AWS CLI (AWS CLI)

1. (Facultatif) Pour afficher le paramètre de durée de session maximale d'un rôle, exécutez la commande suivante :

   - aws iam get-role

2. Pour mettre à jour le paramètre de durée de session maximale d'un rôle, exécutez la commande suivante avec le paramètre `max-session-duration` de l'interface de ligne de commande ou le paramètre `MaxSessionDuration` de l'API :

   - aws iam update-role

   Vos modifications ne prennent pas effet jusqu'à ce qu'une autre personne endosse ce rôle. Pour apprendre à révoquer des sessions existantes pour ce rôle, consultez Révocation des informations d'identification de sécurité temporaires d'un rôle IAM.

Modification d'une limite d'autorisations de rôle (AWS CLI)

Pour modifier le nombre maximum d'autorisations permises pour un rôle, modifiez la limite d'autorisations du rôle

Pour modifier la politique gérée utilisée afin de définir la limite d'autorisations pour un rôle (AWS CLI)

1. (Facultatif) Pour afficher la limite d'autorisations actuelle d'un rôle, exécutez la commande suivante :

   - aws iam get-role

2. Pour utiliser une autre politique gérée afin de mettre à jour la limite d'autorisations d'un rôle, exécutez la commande suivante :

   - était un objectif put-role-permissions-boundary

   Un rôle peut disposer d'une seule politique gérée définie comme une limite d'autorisations. Si vous modifiez la limite d'autorisations, cela modifie le nombre maximum d'autorisations permises pour un rôle.

Modifier un rôle (AWS API)

Vous pouvez utiliser l' AWS API pour modifier un rôle. Pour modifier l'ensemble des balises sur un rôle, consultez Gestion des balises sur les rôles IAM (AWS CLI ou AWS API).

Rubriques

- Modifier une politique de confiance dans les rôles (AWS API)
- Modification d'une politique d'autorisation de rôle (AWS API)
- Modification d'une description de rôle (API AWS)
- Modifier la durée maximale de session (AWS API) d'un rôle
- Modifier la limite des autorisations d'un rôle (AWS API)

Modifier une politique de confiance dans les rôles (AWS API)

Pour modifier la personne qui peut endosser un rôle, vous devez modifier la politique d'approbation du rôle. Vous ne pouvez pas modifier la politique d'approbation d'un rôle lié à un service.

Remarques

- Si un utilisateur est répertorié comme principal dans la politique d'approbation d'un rôle, mais ne peut pas endosser le rôle, vérifiez les limites d'autorisations de l'utilisateur. Si une limite d'autorisation est définie pour l'utilisateur, elle doit autoriser l'action `sts:AssumeRole`.
- Pour permettre aux utilisateurs d'assumer à nouveau le rôle actuel au cours d'une session de rôle, spécifiez l'ARN du rôle ou l' Compte AWS ARN comme principal dans la politique de confiance des rôles. Services AWS qui fournissent des ressources de calcul telles qu'Amazon EC2, Amazon ECS, Amazon EKS et Lambda fournissent des informations d'identification temporaires et mettent automatiquement à jour ces informations d'identification. Cela garantit que vous disposez toujours d'un ensemble d'informations d'identification valide. Pour ces services, il n'est pas nécessaire d'endosser à nouveau le rôle actuel pour obtenir des informations d'identification temporaires. Toutefois, si vous avez l'intention de transférer des balises de session ou une politique de session, vous devez endosser à nouveau le rôle actuel. Pour savoir comment modifier une politique d'approbation des rôles afin d'ajouter l'ARN ou Compte AWS l'ARN du rôle principal, consultezModification d'une politique d'approbation de rôle (console).

Pour modifier une politique de confiance des rôles (AWS API)

1. (Facultatif) Si vous ne connaissez pas le nom du rôle que vous souhaitez modifier, appelez l'opération suivante pour répertorier les rôles de votre compte :
   - ListRoles
2. (Facultatif) Pour afficher la politique d'approbation actuelle d'un rôle, appelez l'opération suivante :
   - GetRole
3. Pour modifier les principaux approuvés ayant accès au rôle, créez un fichier texte avec la politique d'approbation mise à jour. Vous pouvez utiliser l'éditeur de texte pour créer la politique.

   Par exemple, la politique de confiance suivante montre comment faire référence à deux Comptes AWS dans l'`Principal`élément. Cela permet aux utilisateurs de deux Comptes AWS distincts d'endosser ce rôle.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {"AWS": [
            "arn:aws:iam::111122223333:root",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "sts:AssumeRole"
    }
}
```

Si vous spécifiez un principal dans un autre compte, l'ajout d'un compte à la politique d'approbation d'un rôle n'est qu'une partie de la procédure d'établissement de la relation d'approbation entre comptes. Par défaut, aucun utilisateur des comptes approuvés ne peut endosser le rôle. L'administrateur du compte nouvellement approuvé doit accorder aux utilisateurs l'autorisation d'endosser le rôle. Pour ce faire, l'administrateur doit créer ou modifier une politique attachée à l'utilisateur pour lui permettre d'accéder à l'action `sts:AssumeRole`. Pour plus d'informations, consultez la procédure suivante ou Octroi d'autorisations à un utilisateur pour endosser un rôle.

4. Pour utiliser le fichier créé afin de mettre à jour la politique de confiance, appelez l'opération suivante :

   - UpdateAssumeRolePolicy

Pour autoriser les utilisateurs d'un compte externe approuvé à utiliser le rôle (AWS API)

Pour plus d'informations sur cette procédure, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle.

1. Créez un fichier JSON contenant une politique d'autorisations qui accorde des autorisations permettant d'endosser le rôle. Par exemple, la politique suivante contient les autorisations nécessaires minimales :

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
```

```
    "Resource": "arn:aws:iam::ACCOUNT-ID-THAT-CONTAINS-ROLE:role/ROLE-NAME"
  }
}
```

Remplacez l'ARN dans l'instruction par celui du rôle que l'utilisateur peut endosser.

2. Appelez l'opération suivante pour charger le fichier JSON contenant la politique de confiance dans IAM :

   - CreatePolicy

   Le résultat de cette opération inclut l'ARN de la politique. Notez cet ARN, car vous aurez besoin de l'utiliser ultérieurement.

3. Décidez à quel utilisateur ou groupe attacher la politique. Si vous ne connaissez pas le nom de l'utilisateur ou du groupe concerné, appelez les opérations suivantes pour répertorier les utilisateurs ou les groupes de votre compte :

   - ListUsers
   - ListGroups

4. Appelez l'une des opérations suivantes pour attacher la politique créée à l'étape précédente à l'utilisateur ou au groupe :

   - API : AttachUserPolicy
   - AttachGroupPolicy

Modification d'une politique d'autorisation de rôle (AWS API)

Pour modifier les autorisations accordées par le rôle, modifiez la stratégie des autorisations du rôle (ou les stratégies). Vous ne pouvez pas modifier la politique d'autorisations d'un rôle lié à un service dans IAM. Vous pouvez modifier la politique d'autorisations dans le service dépendant du rôle. Pour vérifier si un service prend en charge cette fonctionnalité, reportez-vous à la rubrique AWS services qui fonctionnent avec IAM et recherchez les services qui possèdent Yes (Oui)dans la colonne Rôles liés à un service. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Pour modifier les autorisations accordées par un rôle (AWS API)

1. (Facultatif) Pour afficher les autorisations actuelles associées à un rôle, appelez les opérations suivantes :

   1. ListRolePoliciespour répertorier les politiques en ligne
   2. ListAttachedRolePoliciespour répertorier les politiques gérées

2. L'opération permettant de mettre à jour les autorisations du rôle varie selon que vous mettez à jour une politique gérée ou une politique en ligne.

   Pour mettre à jour une politique gérée, appelez l'opération suivante pour créer une nouvelle version de la politique gérée :

   - CreatePolicyVersion

   Pour mettre à jour une politique en ligne, appelez l'opération suivante :

   - PutRolePolicy

Modification d'une description de rôle (API AWS)

Pour modifier la description du rôle, modifiez le texte de la description.

Pour modifier la description d'un rôle (AWS API)

1. (Facultatif) Pour afficher la description actuelle d'un rôle, appelez l'opération suivante :

   - GetRole

2. Pour mettre à jour la description d'un rôle, appelez l'opération suivante avec le paramètre de description :

   - UpdateRole

Modifier la durée maximale de session (AWS API) d'un rôle

Pour spécifier le paramètre de durée de session maximale pour les rôles qui sont endossés à l'aide de l’interface AWS CLI ou de l'API, modifiez la valeur du paramètre de durée de session maximale. La valeur de ce paramètre peut varier de 1 heure à 12 heures. Si vous ne spécifiez pas de valeur, la

valeur par défaut de 1 heure maximum est appliquée. Ce paramètre ne limite pas les sessions prises en charge par AWS les services.

> **Note**
>
> Toute personne assumant le rôle depuis l'API AWS CLI or peut utiliser le paramètre `duration-seconds` CLI ou le paramètre `DurationSeconds` API pour demander une session plus longue. Le paramètre `MaxSessionDuration` détermine la durée maximale de la session de rôle qui peut être demandée à l'aide du paramètre `DurationSeconds`. Si les utilisateurs ne spécifient pas de valeur pour le paramètre `DurationSeconds`, leurs informations d'identification de sécurité sont valides pendant une heure.

Pour modifier le paramètre de durée maximale de session pour les rôles assumés à l'aide de l'API (AWS API)

1. (Facultatif) Pour afficher le paramètre de durée de session maximale d'un rôle, appelez l'opération suivante :

   - GetRole

2. Pour mettre à jour le paramètre de durée de session maximale d'un rôle, appelez l'opération suivante avec le paramètre `max-sessionduration` de la CLI ou le paramètre `MaxSessionDuration` de l'API :

   - UpdateRole

   Vos modifications ne prennent pas effet jusqu'à ce qu'une autre personne endosse ce rôle. Pour apprendre à révoquer des sessions existantes pour ce rôle, consultez Révocation des informations d'identification de sécurité temporaires d'un rôle IAM.

Modifier la limite des autorisations d'un rôle (AWS API)

Pour modifier le nombre maximum d'autorisations permises pour un rôle, modifiez la limite d'autorisations du rôle

Pour modifier la politique gérée utilisée afin de définir la limite d'autorisations pour un rôle (API AWS)

1. (Facultatif) Pour afficher la limite d'autorisations actuelle d'un rôle, appelez l'opération suivante :

- GetRole

2. Pour utiliser une autre politique gérée afin de mettre à jour la limite d'autorisations d'un rôle, appelez l'opération suivante :

   - PutRolePermissionsBoundary

   Un rôle peut disposer d'une seule politique gérée définie comme une limite d'autorisations. Si vous modifiez la limite d'autorisations, cela modifie le nombre maximum d'autorisations permises pour un rôle.

Suppression de rôles ou de profils d'instance

Si vous n'avez plus besoin d'un rôle, nous vous recommandons de supprimer le rôle et ses autorisations associées. De cette façon, vous n’avez aucune entité inutilisée qui n’est pas surveillée ou gérée activement.

Si le rôle était associé à une instance EC2, vous pouvez également le supprimer du profil d'instance, puis supprimer ce dernier.

> **Warning**
>
> Vérifiez qu'aucune instance Amazon EC2 n'est en cours d'exécution avec le rôle ou le profil d'instance que vous êtes sur le point de supprimer. La suppression d'un rôle ou d'un profil d'instance associé à une instance en cours d'exécution arrêtera toutes les applications qui s'exécutent sur l'instance.

Si vous préférez ne pas supprimer définitivement un rôle, vous pouvez le désactiver. Pour ce faire, modifiez les politiques du rôle, puis révoquez toutes les sessions en cours. Par exemple, vous pouvez ajouter une politique au rôle qui refuse l'accès à tous AWS. Vous pouvez également modifier la politique d'approbation pour refuser l'accès à toute personne qui tente d'endosser le rôle. Pour en savoir plus sur la révocation des sessions, consultez Révocation des informations d'identification de sécurité temporaires d'un rôle IAM.

Rubriques

- Afficher l'accès au rôle
- Suppression d'un rôle lié à un service

- Suppression d'un rôle IAM (console)
- Suppression d'un rôle IAM (AWS CLI)
- Suppression d'un rôle IAM (API AWS)
- Informations connexes

Afficher l'accès au rôle

Avant de supprimer un rôle, nous vous recommandons de vérifier la date de sa dernière utilisation. Vous pouvez le faire à l'aide de l'API AWS Management Console AWS CLI, du ou de l' AWS API. Vous devriez consulter ces informations, car vous ne voulez pas retirer l'accès à une personne utilisant ce rôle.

La date de la dernière activité du rôle peut ne pas correspondre à la dernière date indiquée dans l'onglet Access Advisor. L'onglet Access Advisor signale l'activité uniquement pour les services autorisés par les politiques d'autorisations du rôle. La date de la dernière activité du rôle inclut la dernière tentative d'accès à un service dans lequel le rôle a été créé AWS.

> **Note**
>
> La période de suivi de la dernière activité d'un rôle et des données Access Advisor correspond aux 400 jours précédents. Cette période peut être plus courte si votre région a commencé à prendre en charge ces fonctionnalités au cours de la dernière année. Le rôle aurait pu être utilisé il y a plus de 400 jours. Pour de plus amples informations sur la période de suivi, veuillez consulter Où AWS suit les dernières informations consultées.

Pour afficher la date à laquelle un rôle a été utilisé pour la dernière fois (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Roles (Rôles).
3. Recherchez la ligne du rôle dont vous souhaitez afficher l'activité. Vous pouvez utiliser le champ de recherche pour affiner les résultats. Affichez la colonne Dernière activité pour voir le nombre de jours écoulés depuis la dernière utilisation du rôle. Si le rôle n'a pas été utilisé pendant la période de suivi, le tableau affiche Aucun.
4. Choisissez le nom du rôle pour afficher plus d'informations. La page Summary (Résumé) du rôle inclut également Last activity (Dernière activité), qui affiche la date à laquelle le rôle a été utilisé

pour la dernière fois. Si le rôle n'a pas été utilisé au cours des 400 derniers jours, la Dernière activité affiche Non consulté pendant la période de suivi.

Pour voir quand un rôle a été utilisé pour la dernière fois (AWS CLI)

`aws iam get-role` - Exécutez cette commande pour renvoyer des informations sur un rôle, y compris l'objet `RoleLastUsed`. Cet objet contient la `LastUsedDate` et la `Region` dans laquelle le rôle a été utilisé pour la dernière fois. Si `RoleLastUsed` est présent mais ne contient pas de valeur, le rôle n'a pas été utilisé pendant la période de suivi.

Pour voir quand un rôle a été utilisé pour la dernière fois (AWS API)

`GetRole` - Appelez cette opération pour renvoyer des informations sur un rôle, y compris l'objet `RoleLastUsed`. Cet objet contient la `LastUsedDate` et la `Region` dans laquelle le rôle a été utilisé pour la dernière fois. Si `RoleLastUsed` est présent mais ne contient pas de valeur, le rôle n'a pas été utilisé pendant la période de suivi.

Suppression d'un rôle lié à un service

Si le rôle est un rôle lié à un service, consultez la documentation du service lié afin de savoir comment supprimer le rôle. Pour afficher les rôles liés à un service de votre compte, allez sur la page IAM Roles (Rôles IAM) de la console. Les rôles liés à un service s'affichent avec (Rôle lié à un service) mentionné dans la colonne Entités de confiance du tableau. Une bannière sur la page Summary (Résumé) d'un rôle indique aussi que le rôle est lié à un service.

Si le service ne contient pas de documentation pour supprimer le rôle lié au service, vous pouvez utiliser la console IAM ou l'API pour supprimer le rôle. AWS CLI Pour plus d'informations, consultez Suppression d'un rôle lié à un service.

Suppression d'un rôle IAM (console)

Lorsque vous utilisez le AWS Management Console pour supprimer un rôle, IAM détache automatiquement les politiques gérées associées au rôle. Il supprime aussi automatiquement toute politique en ligne associée au rôle, ainsi que tout profil d'instance Amazon EC2 qui contient le rôle.

> **Important**
>
> Dans certains cas, un rôle peut être associé à un profil d'instance Amazon EC2, et le rôle et le profil d'instance peuvent porter exactement le même nom. Dans ce cas, vous pouvez

utiliser le AWS Management Console pour supprimer le rôle et le profil d'instance. Ce lient se fait automatiquement pour les rôles et les profils d'instance que vous créez dans la console. Si vous avez créé le rôle à partir des outils pour Windows PowerShell ou de l' AWS API, le rôle et le profil d'instance peuvent porter des noms différents. AWS CLI Dans ce cas, vous ne pouvez pas utiliser la console pour les supprimer. Vous devez plutôt utiliser les AWS CLI outils pour Windows PowerShell ou l' AWS API pour supprimer d'abord le rôle du profil d'instance. Vous devez ensuite réaliser une étape distincte pour supprimer le rôle.

Pour supprimer un rôle (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Rôles, puis cochez la case en regard du rôle à supprimer.
3. En haut de la page, sélectionnez Delete (Supprimer).
4. Dans la boîte de dialogue de confirmation, passez en revue les dernières informations consultées, qui indiquent la date à laquelle chacun des rôles sélectionnés a accédé pour la dernière fois à un AWS service. Cela vous permet de confirmer si le rôle est actif actuellement. Pour poursuivre, saisissez le nom du rôle dans le champ de saisie de texte et choisissez Delete (Supprimer). Si vous êtes sûr, procédez à la suppression même si les dernières informations consultées sont toujours en cours de chargement.

> **Note**
>
> Vous ne pouvez pas utiliser la console pour supprimer un profil d'instance, sauf lorsqu'elle porte le même nom que le rôle. Le profil d'instance est supprimé dans le cadre du processus de suppression d'un rôle comme décrit dans la procédure précédente. Pour supprimer un profil d'instance sans supprimer également le rôle, vous devez utiliser l' AWS API AWS CLI or. Pour plus d'informations, consultez les sections suivantes.

Suppression d'un rôle IAM (AWS CLI)

Lorsque vous utilisez le AWS CLI pour supprimer un rôle, vous devez d'abord supprimer les politiques intégrées associées au rôle. Vous devez également détacher les politiques gérées

associées au rôle. Si vous voulez supprimer le profil d'instance associé contenant le rôle, vous devez le supprimer séparément.

Pour supprimer un rôle (AWS CLI)

1. Si vous ne connaissez pas le nom du rôle que vous souhaitez supprimer, tapez la commande suivante pour répertorier les rôles dans votre compte :

   ```
   aws iam list-roles
   ```

   La liste inclut l'Amazon Resource Name (ARN) de chaque rôle. Utilisez le nom du rôle, pas l'ARN, pour faire référence aux rôles avec les commandes CLI. Par exemple, si un rôle a l'ARN : `arn:aws:iam::123456789012:role/myrole`, vous faites référence au rôle en tant que **`myrole`**.
2. Supprimez le rôle de tous les profils d'instance auxquels il est associé.

   a. Pour répertorier tous les profils d'instance auxquels un rôle est associé, tapez la commande suivante :

      ```
      aws iam list-instance-profiles-for-role --role-name role-name
      ```

   b. Pour supprimer le rôle d'un profil d'instance, saisissez la commande suivante pour chaque profil instance :

      ```
      aws iam remove-role-from-instance-profile --instance-profile-name instance-profile-name --role-name role-name
      ```
3. Supprimez toutes les politiques associées au rôle.

   a. Pour répertorier toutes les politiques en ligne que contient le rôle, saisissez la commande suivante :

      ```
      aws iam list-role-policies --role-name role-name
      ```

   b. Pour supprimer chaque politique en ligne du rôle, saisissez la commande suivante pour chaque politique :

      ```
      aws iam delete-role-policy --role-name role-name --policy-name policy-name
      ```

c. Pour répertorier toutes les politiques gérées attachées au rôle, saisissez la commande suivante :

```
aws iam list-attached-role-policies --role-name role-name
```

d. Pour détacher chaque politique gérée du rôle, saisissez la commande suivante pour chaque politique :

```
aws iam detach-role-policy --role-name role-name --policy-arn policy-arn
```

4. Tapez la commande suivante pour supprimer le rôle :

```
aws iam delete-role --role-name role-name
```

5. Si vous ne prévoyez pas de réutiliser les profils d'instance associés au rôle, vous pouvez saisir la commande suivante pour les supprimer :

```
aws iam delete-instance-profile --instance-profile-name instance-profile-name
```

Suppression d'un rôle IAM (API AWS)

Lorsque vous utilisez l'API IAM pour supprimer un rôle, vous devez d'abord supprimer les politiques en ligne associées au rôle. Vous devez également détacher les politiques gérées associées au rôle. Si vous voulez supprimer le profil d'instance associé contenant le rôle, vous devez le supprimer séparément.

Pour supprimer un rôle (AWS API)

1. Pour répertorier tous les profils d'instance auxquels un rôle est associé, appelez ListInstanceProfilesForRole.

   Pour supprimer le rôle d'un profil d'instance, appelez RemoveRoleFromInstanceProfile. Vous devez transmettre le nom du rôle et le nom du profil d'instance.

   Si vous ne comptez pas réutiliser un profil d'instance associé au rôle, appelez DeleteInstanceProfilepour le supprimer.
2. Pour répertorier toutes les politiques intégrées à un rôle, appelez ListRolePolicies.

Pour supprimer les politiques intégrées associées au rôle, appelez DeleteRolePolicy. Vous devez transmettre le nom du rôle et le nom de la politique en ligne.

3. Pour répertorier toutes les politiques gérées associées à un rôle, appelez ListAttachedRolePolicies.

   Pour détacher les politiques gérées associées au rôle, appelez DetachRolePolicy. Vous devez transmettre le nom du rôle et l'ARN de la politique gérée.

4. Appelez DeleteRolepour supprimer le rôle.

Informations connexes

Pour obtenir des informations générales sur les profils d'instance, consultez Utilisation de profils d'instance.

Pour obtenir des informations générales sur les rôles liés à un service, veuillez consulter Utilisation des rôles liés à un service.

Fournisseurs d'identité et fédération

Si vous gérez déjà les identités des utilisateurs en dehors de AWS, vous pouvez utiliser des fournisseurs d'identité au lieu de créer des utilisateurs IAM dans votre Compte AWS. Avec un fournisseur d'identité (IdP), vous pouvez gérer les identités de vos utilisateurs en dehors de votre AWS compte et leur donner l'autorisation d'utiliser les AWS ressources de votre compte. Ceci est utile si votre organisation dispose déjà de son propre système d'identité, par exemple un répertoire d'utilisateurs d'entreprise. Il en est de même si vous créez une application web ou mobile devant accéder aux ressources AWS .

Un IdP externe fournit des informations d'identité à l' AWS aide d'OpenID Connect (OIDC) ou de SAML 2.0 (Security Assertion Markup Language 2.0). L'OIDC connecte des applications, telles que GitHub Actions, qui ne s'exécutent pas sur AWS des AWS ressources. Shibboleth et Active Directory Federation Services sont des exemples de fournisseurs d'identité SAML bien connus.

> **Note**
>
> En tant que bonne pratique de sécurité, nous vous recommandons de gérer les utilisateurs humains dans l'IAM Identity Center avec un fournisseur d'identité SAML externe plutôt que d'utiliser la fédération SAML dans IAM. Pour en savoir plus sur les situations spécifiques dans

lesquelles un utilisateur IAM est nécessaire, veuillez consulter Quand créer un utilisateur IAM (au lieu d'un rôle).

Lorsque vous utilisez un fournisseur d'identité , vous n'avez pas à créer de code de connexion personnalisé ni à gérer vos propres identités utilisateur. L'IdP prévoit cela pour vous. Vos utilisateurs externes se connectent via un IdP, et vous pouvez autoriser ces identités externes à utiliser les AWS ressources de votre compte. Les fournisseurs d'identité contribuent à votre Compte AWS sécurité car vous n'avez pas à distribuer ou à intégrer des informations de sécurité à long terme, telles que des clés d'accès, dans votre application.

Ce guide aborde la fédération IAM. Votre cas d'utilisation pourrait être mieux pris en charge par IAM Identity Center ou Amazon Cognito. Les résumés et le tableau suivants fournissent une vue d'ensemble des méthodes que vos utilisateurs peuvent utiliser pour obtenir un accès fédéré aux AWS ressources.

	Account type (Type de compte)	Gestion de l'accès de…	Source d'identité prise en charge
Fédération avec IAM Identiy Center	Plusieurs comptes gérés par AWS Organizations	Les utilisateurs humains de votre personnel	• SAML 2.0 • Managed Active Directory • Répertoire d'Identity Center
Fédération avec IAM	Compte unique et autonome	• Utilisateurs humains dans le cadre de déploiements à court terme et à petite échelle • Utilisateurs machines	• SAML 2.0 • OIDC
Fédération avec les réserves d'identités Amazon Cognito	N’importe quel compte	Les utilisateurs d'applications qui nécessitent une autorisation IAM	• SAML 2.0 • OIDC • Sélectionner les fournisseurs

	Account type (Type de compte)	Gestion de l'accès de…	Source d'identité prise en charge
		pour accéder aux ressources	d'identité sociaux OAuth 2.0

Fédération avec IAM Identiy Center

Pour une gestion centralisée des accès des utilisateurs humains, nous vous recommandons d'utiliser IAM Identity Center pour gérer l'accès à vos comptes et les autorisations au sein de ceux-ci. Les utilisateurs d'IAM Identity Center reçoivent des informations d'identification à court terme pour vos AWS ressources. Vous pouvez utiliser Active Directory, un fournisseur d'identité externe (IdP) ou un annuaire IAM Identity Center comme source d'identité permettant aux utilisateurs et aux groupes d'attribuer l'accès à vos ressources. AWS

IAM Identity Center prend en charge la fédération des identités avec le langage SAML (Security Assertion Markup Language) 2.0 afin de fournir un accès d'authentification unique fédéré aux utilisateurs autorisés à utiliser les applications du portail d'accès. AWS Les utilisateurs peuvent ensuite se connecter de manière unique aux services compatibles SAML, notamment aux applications AWS Management Console et aux applications tierces, telles que Microsoft 365, SAP Concur et Salesforce.

Fédération avec IAM

Bien que nous recommandions fortement de gérer les utilisateurs humains dans IAM Identity Center, vous pouvez activer l'accès utilisateur fédéré avec IAM pour les utilisateurs humains dans le cadre de déploiements à court terme et à petite échelle. IAM vous permet d'utiliser des protocoles SAML 2.0 et Open ID Connect (OIDC) distincts IdPs et d'utiliser des attributs utilisateur fédérés pour le contrôle d'accès. Avec IAM, vous pouvez transmettre des attributs utilisateur, tels que le centre de coûts, le titre ou les paramètres régionaux, de votre compte IdPs à AWS, et mettre en œuvre des autorisations d'accès détaillées en fonction de ces attributs.

Une charge de travail est un ensemble de ressources et de code qui fournit une valeur business, par exemple une application destinée au client ou un processus de backend. Votre charge de travail peut nécessiter une identité IAM pour envoyer des demandes aux AWS services, aux applications, aux outils opérationnels et aux composants. Ces identités incluent les machines exécutées dans vos AWS environnements, telles que les instances ou AWS Lambda les fonctions Amazon EC2.

Vous pouvez également gérer les Identités machine pour les parties externes qui ont besoin d'un accès. Pour donner accès aux identités des machines, vous pouvez utiliser des rôles IAM. Les rôles IAM disposent d'autorisations spécifiques et fournissent un moyen d'accès AWS en s'appuyant sur des informations d'identification de sécurité temporaires associées à une session de rôle. En outre, il se peut AWS que des machines extérieures aient besoin d'accéder à vos AWS environnements. Pour les machines qui s'exécutent en dehors de AWS vous, vous pouvez utiliser IAM Roles Anywhere. Pour plus d'informations sur les rôles , consultez Rôles IAM. Pour plus de détails sur l'utilisation des rôles pour déléguer l'accès entre Comptes AWS eux, consultezDidacticiel IAM : déléguer l'accès entre des comptes AWS à l'aide des rôles IAM.

Pour lier un IdP directement à IAM, vous devez créer une entité fournisseur d'identité afin d'établir une relation de confiance entre vous et Compte AWS l'IdP. Les supports IAM sont IdPs compatibles avec OpenID Connect (OIDC) ou SAML 2.0 (Security Assertion Markup Language 2.0). Pour plus d'informations sur l'utilisation de l'un d' IdPs entre eux avec AWS, consultez les sections suivantes :

- Fédération OIDC
- Fédération SAML 2.0

Fédération avec les réserves d'identités Amazon Cognito

Amazon Cognito est conçu pour les développeurs qui souhaitent authentifier et autoriser les utilisateurs dans leurs applications mobiles et Web. Les groupes d'utilisateurs Amazon Cognito ajoutent des fonctionnalités de connexion et d'inscription à votre application, et les réserves d'identités fournissent des informations d'identification IAM qui permettent à vos utilisateurs d'accéder aux ressources protégées que vous gérez dans AWS. Les réserves d'identités obtiennent des informations d'identification pour les sessions temporaires par le biais de l'opération API `AssumeRoleWithWebIdentity`.

Amazon Cognito travaille avec des fournisseurs d'identité externes qui prennent en charge SAML et OpenID Connect, ainsi qu'avec des fournisseurs d'identité sociaux tels que Facebook, Google et Amazon. Votre application peut connecter un utilisateur appartenant à un groupe d'utilisateurs ou à un IdP externe, puis récupérer des ressources en son nom grâce à des sessions temporaires personnalisées dans un rôle IAM.

Scénarios courants

> **Note**
>
> Nous vous recommandons de demander à vos utilisateurs humains d'utiliser des informations d'identification temporaires lors de l'accès AWS. Avez-vous envisagé d'en utiliser AWS IAM Identity Center ? Vous pouvez utiliser IAM Identity Center pour gérer de manière centralisée l'accès à plusieurs comptes Comptes AWS et fournir aux utilisateurs un accès par authentification unique protégé par le MFA à tous les comptes qui leur sont attribués à partir d'un seul endroit. Avec IAM Identity Center, vous pouvez créer et gérer les identités des utilisateurs dans IAM Identity Center ou vous connecter facilement à votre fournisseur d'identité compatible SAML 2.0 existant. Pour plus d'informations, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Vous pouvez utiliser un fournisseur d'identité (IdP) externe pour gérer les identités des utilisateurs en dehors de... et de l'IdP externe AWS. Un IdP externe peut fournir des informations d'identité à l' AWS aide d'OpenID Connect (OIDC) ou du langage SAML (Security Assertion Markup Language). L'OIDC est couramment utilisé lorsqu'une application qui ne s'exécute pas AWS a besoin d'accéder à AWS des ressources.

Lorsque vous souhaitez configurer la fédération avec un IdP externe, vous créez un fournisseur d'identité IAM pour fournir des AWS informations sur l'IdP externe et sa configuration. Cela établit la confiance entre votre Compte AWS IdP et l'IdP externe. Les rubriques suivantes présentent des scénarios courants d'utilisation des fournisseurs d'identité IAM.

Rubriques

- Utilisation d'Amazon Cognito pour les applications mobiles
- Utilisation des opérations de l'API de fédération OIDC pour les applications mobiles

Utilisation d'Amazon Cognito pour les applications mobiles

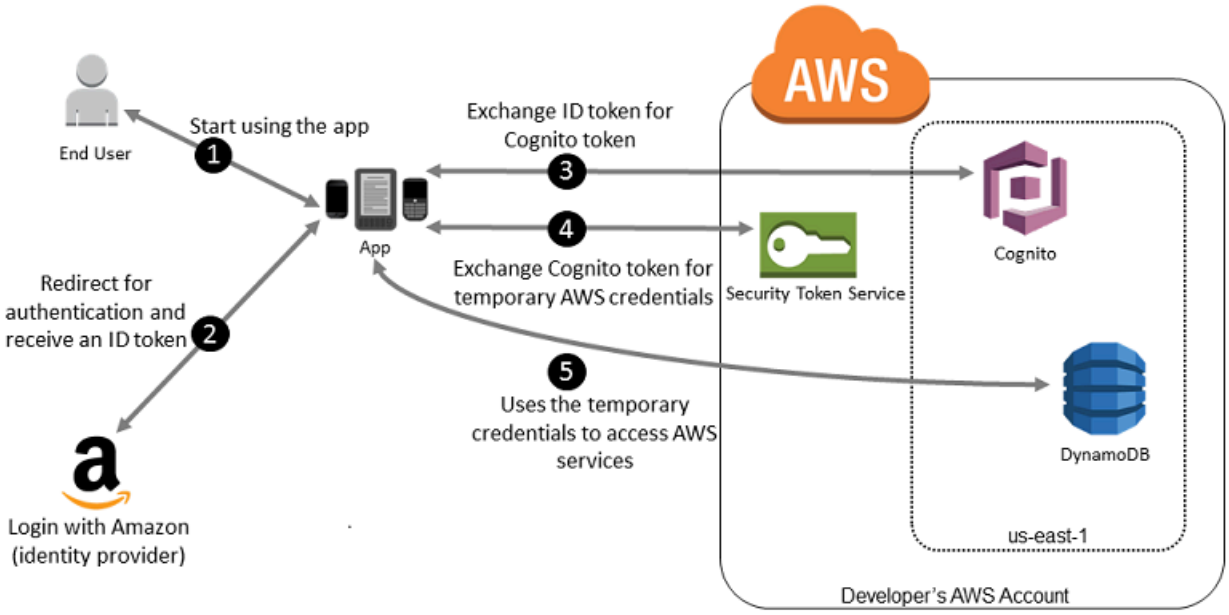
La méthode préférée pour utiliser la fédération OIDC est d'utiliser Amazon Cognito. Par exemple, la développeuse Adèle est en train de créer un jeu pour appareil mobile dans lequel les données utilisateur, telles que les scores et les profils, sont stockées dans Amazon S3 et Amazon DynamoDB. Adèle pourrait également stocker ces données en local sur l'appareil et utiliser Amazon Cognito pour les maintenir synchronisées sur les différents appareils. Elle sait que pour des raisons de sécurité

et de maintenance, des informations d'identification (credentials) de sécurité AWS à long terme ne doivent pas être distribuées avec le jeu. Elle sait également que le jeu pourrait avoir un grand nombre d'utilisateurs. Pour toutes ces raisons, elle ne veut pas créer de nouvelles identités dans IAM pour chaque joueur. Au lieu de cela, elle développe le jeu de sorte que les utilisateurs puissent se connecter à l'aide d'une identité qu'ils ont déjà établie avec un fournisseur d'identité (IdP) externe connu, comme Login with Amazon, Facebook, Google, ou tout fournisseur d'identité compatible OpenID Connect (OIDC). Son jeu peut tirer parti du mécanisme d'authentification de l'un de ces fournisseurs afin de valider l'identité de l'utilisateur.

Pour permettre à l'application mobile d'accéder à ses AWS ressources, Adele enregistre d'abord un identifiant de développeur auprès de son choix IdPs. Elle configure également l'application avec chacun de ces fournisseurs. Dans le dossier Compte AWS qui contient le compartiment Amazon S3 et la table DynamoDB du jeu, Adele utilise Amazon Cognito pour créer des rôles IAM qui définissent précisément les autorisations dont le jeu a besoin. Si elle utilise un IdP OIDC, elle crée également une entité de fournisseur d'identité IAM OIDC pour établir un lien de confiance entre le pool d'identités Amazon Cognito qu'elle possède et l'IdP. Compte AWS

Dans le code de l'application, Adèle appelle l'interface de la connexion pour le fournisseur d'identité l'IdP qu'elle a configuré précédemment. Le fournisseur d'identité traite tous les détails permettant à l'utilisateur de se connecter, et l'application obtient un jeton d'accès OAuth ou jeton d'ID OIDC du fournisseur. L'application d'Adele peut échanger ces informations d'authentification contre un ensemble d'informations de sécurité temporaires comprenant un identifiant de clé d' AWS accès, une clé d'accès secrète et un jeton de session. L'application peut ensuite utiliser ces informations d'identification pour accéder aux services Web proposés par AWS. L'application est limitée aux autorisations qui sont définies dans le rôle qu'elle endosse.

La figure suivante illustre un flux simplifié d'un fonctionnement possible, à l'aide de Login with Amazon comme fournisseur d'identité. Pour l'étape 2, l'application peut également utiliser Facebook, Google ou tout fournisseur d'identité OIDC compatible, mais ce n'est pas présenté ici.



1. Un client démarre votre application sur un appareil mobile. L'application demande à l'utilisateur de se connecter.
2. L'application utilise des ressources Login with Amazon pour accepter les informations d'identification de l'utilisateur.
3. L'application utilise des opérations d'API Amazon Cognito `GetId` et `GetCredentialsForIdentity` pour échanger le jeton d'ID Login with Amazon (Connexion avec Amazon) contre un jeton Amazon Cognito. Amazon Cognito, qui a été configuré pour approuver votre projet Login with Amazon (Connexion avec Amazon), génère un jeton qu'il échange contre des informations d'identification de session temporaires avec AWS STS.
4. L'application reçoit des informations d'identification de sécurité temporaires d'Amazon Cognito. Votre application peut également utiliser le flux de travail de base (classique) d'Amazon Cognito pour récupérer des jetons lors de AWS STS l'utilisation. `AssumeRoleWithWebIdentity` Pour plus d'informations, consultez la rubrique Flux d'authentification des groupes d'identités (identités fédérées) dans le Guide du développeur Amazon Cognito.
5. Les informations d'identification de sécurité temporaires peuvent être utilisées par l'application pour accéder aux ressources AWS requises par l'application pour fonctionner. Le rôle associé aux informations d'identification de sécurité temporaires et les politiques qui lui sont attribuées détermine ce qui est accessible.

Suivez le processus suivant pour configurer votre application afin qu'elle utilise Amazon Cognito afin d'authentifier les utilisateurs et de permettre à votre application d'accéder aux ressources. AWS Pour les étapes spécifiques permettant de réaliser ce scénario, consultez la documentation d'Amazon Cognito.

1. (Facultatif) Connectez-vous en tant que développeur auprès de Login with Amazon, Facebook, Google ou tout autre fournisseur d'identité compatible OpenID Connect (OIDC) et configurez une ou plusieurs applications avec le fournisseur. Cette étape est facultative, car Amazon Cognito prend également en charge l'accès non authentifié (invité) pour vos utilisateurs.
2. Accédez à Amazon Cognito dans le. AWS Management Console Utilisez l'assistant Amazon Cognito pour créer un groupe d'identités, qui est un conteneur utilisé par Amazon Cognito pour maintenir des identités d'utilisateur final organisées pour vos applications. Vous pouvez partager des pools d'identité entre des applications. Lorsque vous configurez un groupe d'identités, Amazon Cognito crée un ou deux rôles IAM (un pour les identités authentifiées et un pour les identités « invitées » non authentifiées) qui définissent des autorisations pour les utilisateurs d'Amazon Cognito.
3. Intégrez AWSAmplify avec votre application et importez les fichiers requis pour utiliser Amazon Cognito.
4. Créez une instance du fournisseur d'informations d'identification Amazon Cognito en transmettant l'ID de groupe d'identité, votre numéro d' Compte AWS et l'Amazon Resource Name (ARN) des rôles que vous avez associés au groupe d'identités. L'assistant Amazon Cognito AWS Management Console fournit un exemple de code pour vous aider à démarrer.
5. Lorsque votre application accède à une AWS ressource, transmettez l'instance du fournisseur d'informations d'identification à l'objet client, qui transmet les informations de sécurité temporaires au client. Les autorisations pour les informations d'identification sont basées sur le ou les rôles que vous avez définis précédemment.

Pour plus d’informations, consultez les ressources suivantes :

- Connectez-vous (Android) dans la documentation du AWS Amplify framework.
- Connectez-vous (iOS) dans la documentation du AWS Amplify framework.

Utilisation des opérations de l'API de fédération OIDC pour les applications mobiles

Pour de meilleurs résultats, utilisez Amazon Cognito comme courtier d'identité pour presque tous les scénarios de fédération OIDC. Amazon Cognito est facile à utiliser et offre des

capacités supplémentaires, comme l'accès anonyme (sans authentification) et la synchronisation des données utilisateur entre les périphériques et les fournisseurs. Toutefois, si vous avez déjà créé une application qui utilise la fédération OIDC en appelant manuellement l'`AssumeRoleWithWebIdentity`API, vous pouvez continuer à l'utiliser et vos applications fonctionneront toujours correctement.

Le processus d'utilisation de la fédération OIDC sans Amazon Cognito suit les grandes lignes suivantes :

1. Inscrivez-vous en tant que développeur auprès de l'IdP externe et configurez votre application avec l'ID unique qu'il vous donnera. (Différents IdPs utilisent une terminologie différente pour ce processus. Ce plan utilise le terme configurer pour désigner le processus d'identification de votre application auprès de l'IdP.) Chaque IdP vous donne un identifiant d'application qui lui est propre. Ainsi, si vous configurez la même application avec plusieurs identifiants IdPs, votre application aura plusieurs identifiants d'application. Vous pouvez configurer plusieurs applications auprès de chaque fournisseur.

   Les liens externes suivants fournissent des informations sur l'utilisation de certains fournisseurs d'identité couramment utilisés (IdPs) :

   - Login with Amazon Developer Center
   - Add Facebook Login to Your App or Website sur le site des développeurs Facebook.
   - Using OAuth 2.0 for Login (OpenID Connect) sur le site des développeurs Google.

   > **Important**
   >
   > Si vous utilisez un fournisseur d'identité OIDC de Google, Facebook ou Amazon Cognito, ne créez pas de fournisseur d'identité IAM distinct dans le. AWS Management Console AWS intègre ces fournisseurs d'identité OIDC et les met à votre disposition. Ignorez l'étape suivante et passez directement à la création de rôles à l'aide de votre fournisseur d'identité.

2. Si vous utilisez un IdP autre que Google, Facebook ou Amazon Cognito, compatible avec OIDC, créez une entité de fournisseur d'identité IAM pour lui.
3. Dans IAM, créez un ou plusieurs rôles. Pour chaque rôle, définissez qui peut endosser le rôle (politique d'approbation) et les autorisations accordées aux utilisateurs de l'application (politique d'autorisation). En général, vous créez un rôle pour chaque IdP pris en charge par l'application. Par exemple, vous pouvez créer un rôle qui est endossé par une application si l'utilisateur se

connecte via Login with Amazon, un deuxième rôle pour la même application s'il se connecte via Facebook, et un troisième rôle s'il se connecte via Google. Pour la relation d'approbation, spécifiez l'IdP (par exemple, Amazon.com) en tant que `Principal` (l'entité de confiance) et incluez un élément `Condition` qui correspond à l'ID d'application attribué par l'IdP. Des exemples de rôles pour différents fournisseurs sont présentés dans la rubrique Création d'un rôle pour un fournisseur d'identité tiers (fédération).

4. Dans votre application, authentifiez vos utilisateurs en recourant à l'IdP. La procédure à suivre varie en fonction de l'IdP utilisé (Login with Amazon, Facebook ou Google) et la plateforme sur laquelle s'exécute votre application. Par exemple, la méthode d'authentification d'une application Android peut être différente de celle d'une application iOS ou d'une application Web JavaScript basée sur le Web.

   En règle générale, si l'utilisateur n'est pas déjà connecté, l'IdP affiche une page de connexion. Une fois qu'il a authentifié l'utilisateur, l'IdP retourne un jeton d'authentification contenant des informations sur l'utilisateur à l'application. Ces informations dépendent de ce que l'IdP expose et des informations que l'utilisateur est disposé à partager. Vous pouvez utiliser ces informations dans votre application.

5. Dans l'application, effectuez un appel non signé à l'action `AssumeRoleWithWebIdentity` pour solliciter des informations d'identification de sécurité temporaires. Dans la demande, vous transmettez le jeton d'authentification de l'IdP et vous spécifiez le nom de ressource Amazon (ARN) pour le rôle IAM que vous avez créé pour cet IdP. AWS vérifie que le jeton est fiable et valide et, dans l'affirmative, renvoie les informations de sécurité temporaires à votre application qui disposent des autorisations pour le rôle que vous nommez dans la demande. La réponse inclut également des métadonnées se rapportant à l'utilisateur provenant de l'IdP telles que l'ID utilisateur unique que l'IdP associe à l'utilisateur.

6. À l'aide des informations de sécurité temporaires fournies dans la `AssumeRoleWithWebIdentity` réponse, votre application envoie des demandes signées aux opérations d' AWS API. Les informations d'identification d'utilisateur fournies par l'IdP permettent de distinguer les utilisateurs de votre application. Par exemple, vous pouvez placer des objets dans des dossiers Amazon S3 qui incluent l'ID utilisateur sous forme de préfixes ou de suffixes. Ceci vous permet de créer des politiques d'accès qui verrouillent un dossier, afin que seul l'utilisateur ayant l'ID approprié soit autorisé à y accéder. Pour plus d’informations, consultez AWS STS principes de session utilisateur fédérée.

7. Votre application met généralement en cache ces informations d'identification de sécurité temporaires afin que vous n'ayez pas à en redemander à chaque fois que l'application doit envoyer une demande à AWS. Par défaut, les informations d'identification restent valides pendant une

heure. Lorsque les informations d'identification expirent (ou avant le délai d'expiration), vous effectuez un autre appel à `AssumeRoleWithWebIdentity` pour obtenir un nouvel ensemble d'informations d'identification de sécurité temporaires. Selon l'IdP utilisé et la façon dont il gère ses jetons, il sera peut-être nécessaire d'actualiser le jeton de l'IdP avant un nouvel appel à `AssumeRoleWithWebIdentity`, car les jetons de l'IdP expirent généralement après un délai spécifié. Si vous utilisez le AWS SDK pour iOS ou le SDK pour Android, vous pouvez utiliser AWS l'action CredentialsProvider AmazonSTS, qui gère les informations d'identification temporaires IAM, notamment en les actualisant si nécessaire.

Fédération OIDC

Imaginez que vous créez une application qui accède à des AWS ressources, comme GitHub Actions, qui utilise des flux de travail pour accéder à Amazon S3 et DynamoDB.

Lorsque vous utilisez ces flux de travail, vous envoyez des demandes aux AWS services qui doivent être signées à l'aide d'une clé d' AWS accès. Cependant, nous vous recommandons vivement de ne pas stocker les AWS informations d'identification à long terme dans des applications extérieures AWS. Configurez plutôt vos applications pour demander des informations d'identification de AWS sécurité temporaires de manière dynamique en cas de besoin à l'aide de la fédération OIDC. Les informations d'identification temporaires fournies correspondent à un AWS rôle qui dispose uniquement des autorisations nécessaires pour effectuer les tâches requises par l'application.

Avec la fédération OIDC, vous n'avez pas besoin de créer de code de connexion personnalisé ni de gérer vos propres identités d'utilisateur. Vous pouvez plutôt utiliser OIDC dans des applications, telles que GitHub Actions ou tout autre IdP compatible avec OpenID Connect (OIDC), pour vous authentifier auprès de. AWS Ils reçoivent un jeton d'authentification, connu sous le nom de jeton Web JSON (JWT), puis échangent ce jeton contre des informations d'identification de sécurité temporaires associées à un rôle IAM autorisé à utiliser des ressources spécifiques dans votre. AWS Compte AWS L'utilisation d'un IdP vous aide à Compte AWS garantir votre sécurité, car vous n'avez pas à intégrer et à distribuer des informations de sécurité à long terme dans votre application.

Pour la plupart des scénarios, il est recommandé d'utiliser Amazon Cognito, car cette application agit en tant que broker d'identité et effectue automatiquement la plupart des tâches liées à la fédération. Pour plus d'informations, consultez la section Utilisation d'Amazon Cognito pour les applications mobiles.

> **Note**
>
> Les jetons Web JSON (JWT) émis par les fournisseurs d'identité OpenID Connect (OIDC) contiennent un délai d'expiration dans la `exp` réclamation qui indique la date d'expiration du jeton. IAM fournit une fenêtre de cinq minutes au-delà du délai d'expiration spécifié dans le JWT pour tenir compte du décalage horaire, comme le permet la norme OpenID Connect (OIDC) Core 1.0. Cela signifie que les JWT OIDC reçus par IAM après le délai d'expiration, mais dans ce délai de cinq minutes, sont acceptés pour une évaluation et un traitement plus approfondis.

Rubriques

- Création d'un fournisseur d'identité OpenID Connect (OIDC) dans IAM
- Obtenir l'empreinte numérique d'un fournisseur d'identité OpenID Connect
- Ressources supplémentaires pour la fédération OIDC

Création d'un fournisseur d'identité OpenID Connect (OIDC) dans IAM

Les fournisseurs d'identité OIDC IAM sont des entités qui, dans IAM, décrivent un service de fournisseur d'identité (IdP) prenant en charge la norme OpenID Connect (OIDC), comme Google ou Salesforce. Vous utilisez un fournisseur d'identité OIDC IAM lorsque vous souhaitez établir une approbation entre un fournisseur d'identité compatible OIDC et votre Compte AWS. Cela est utile lorsque vous créez une application mobile ou une application Web qui nécessite un accès à AWS des ressources, mais vous ne souhaitez pas créer de code de connexion personnalisé ou gérer vos propres identités d'utilisateur. Pour plus d'informations sur ce scénario, consultez the section called “Fédération OIDC”.

Vous pouvez créer et gérer un fournisseur d'identité IAM OIDC à l' AWS Management Console aide des AWS Command Line Interface outils pour Windows ou de l' PowerShellAPI IAM.

Après avoir créé un fournisseur d'identité OIDC IAM, vous devez créer un ou plusieurs rôles IAM. Un rôle est une identité AWS qui ne possède pas ses propres informations d'identification (comme le fait un utilisateur). Mais dans ce contexte, un rôle est attribué dynamiquement à un utilisateur fédéré qui est authentifié par le fournisseur d'identité (IdP) de votre organisation. Le rôle permet à l'IdP de votre organisation de demander des informations d'identification de sécurité temporaires pour accéder à AWS. Les politiques attribuées au rôle déterminent ce que les utilisateurs fédérés sont autorisés à

faire dans AWS ce rôle. Pour créer un rôle pour un fournisseur d'identité tiers, consultez la section Création d'un rôle pour un fournisseur d'identité tiers (fédération).

> **Important**
>
> Lorsque vous configurez des politiques IAM basées sur l'identité pour les actions qui prennent en charge les ressources `oidc-provider`, IAM évalue l'URL complète du fournisseur d'identité OIDC, y compris les chemins spécifiés. Si l'URL de votre fournisseur d'identité OIDC comporte un chemin, vous devez inclure ce chemin dans l'ARN `oidc-provider` en tant que valeur de l'élément `Resource`. Vous avez également la possibilité d'ajouter une barre oblique et un caractère générique (`/*`) au domaine de l'URL ou d'utiliser des caractères génériques (`*` et `?`) à n'importe quel endroit du chemin de l'URL. Si l'URL du fournisseur d'identité OIDC dans la demande ne correspond pas à la valeur définie dans l'élément `Resource` de la politique, la demande échoue.

Pour résoudre les problèmes courants liés à la fédération IAM OIDC, consultez la section Résoudre les erreurs liées à OIDC sur Re:post. AWS

Rubriques

- Conditions préalables : Valider la configuration de votre fournisseur d'identité
- Création et gestion d'un fournisseur OIDC (console)
- Création et gestion d'un fournisseur d'identité OIDC IAM (AWS CLI)
- Création et gestion d'un fournisseur d'identité OIDC (AWS API)

Conditions préalables : Valider la configuration de votre fournisseur d'identité

Avant de créer un fournisseur d'identité IAM OIDC, vous devez disposer des informations suivantes auprès de votre IdP. Pour plus d'informations sur l'obtention des informations de configuration du fournisseur OIDC, consultez la documentation de votre IdP.

1. Déterminez l'URL accessible au public de votre fournisseur d'identité OIDC. L'URL doit commencer par https://. Selon la norme OIDC, les composants du chemin sont autorisés, mais pas les paramètres de requête. Généralement, l'URL se compose uniquement d'un nom d'hôte, tel que https://server.example.org ou https://example.com. L'URL ne doit pas contenir de numéro de port.

2. Ajoutez /.well-known/openid-configuration à la fin de l'URL de votre fournisseur d'identité OIDC pour voir le document de configuration et les métadonnées accessibles au public du fournisseur. Vous devez disposer d'un document de découverte au format JSON contenant le document de configuration du fournisseur et les métadonnées qui peuvent être récupérées à partir de l'URL du point de terminaison de découverte du fournisseur OpenID Connect.
3. Vérifiez que les valeurs suivantes sont incluses dans les informations de configuration de votre fournisseur. Si l'un de ces champs est absent de votre configuration openid, vous devez mettre à jour votre document de découverte. Ce processus peut varier en fonction de votre fournisseur d'identité. Suivez donc la documentation de votre IdP pour effectuer cette tâche.

   - émetteur : URL de votre domaine.
   - jwks_uri : point de terminaison JSON Web Key Set (JWKS) où IAM obtient vos clés publiques. Votre fournisseur d'identité doit inclure un point de terminaison JSON Web Key Set (JWKS) dans la configuration openid. Cette URI définit où obtenir les clés publiques utilisées pour vérifier les jetons signés auprès de votre fournisseur d'identité.
   - claims_supported : informations sur l'utilisateur qui vous aident à garantir que les réponses d'authentification OIDC de votre IdP contiennent les attributs requis AWS utilisés dans les politiques IAM pour vérifier les autorisations des utilisateurs fédérés. Pour obtenir la liste des clés de condition IAM pouvant être utilisées pour les réclamations, consultezClés disponibles pour la AWS fédération OIDC.
     - aud : Vous devez déterminer la valeur d'audience déclarée par votre IdP en JSON Web Tokens (JWT). La réclamation d'audience (aud) est spécifique à l'application et identifie les destinataires prévus du jeton. Lorsque vous enregistrez une application mobile ou Web auprès d'un fournisseur OpenID Connect, celui-ci établit un identifiant client qui identifie l'application. L'identifiant client est un identifiant unique pour votre application qui est transmis dans la demande d'authentification. La réclamation AUD doit correspondre à la valeur Audience lors de la création de votre fournisseur d'identité IAM OIDC.
     - iat : Les réclamations doivent inclure une valeur représentant l'heure à laquelle le jeton d'identification est émis. `iat`
     - iss : URL du fournisseur d'identité. L'URL doit commencer par https ://et doit correspondre à l'URL du fournisseur fournie à IAM. Selon la norme OIDC, les composants du chemin sont autorisés, mais pas les paramètres de requête. Généralement, l'URL se compose uniquement d'un nom d'hôte, tel que https://server.example.org ou https://example.com. L'URL ne doit pas contenir de numéro de port.
   - réponse_types_supported : id_token

- subject_types_supported : public
- id_token_signing_alg_values_supported : RS256

> **Note**
>
> Vous pouvez inclure des réclamations supplémentaires, telles que des réclamations personnalisées, dans l'exemple ci-dessous ; toutefois, AWS STS vous ignorerez la réclamation.

```
{
  "issuer": "https://example-domain.com",
  "jwks_uri": "https://example-domain.com/jwks/keys",
  "claims_supported": [
    "aud",
    "iat",
    "iss",
    "name",
    "sub",
    "custom"
  ],
  "response_types_supported": [
    "id_token"
  ],
  "id_token_signing_alg_values_supported": [
    "RS256"
  ],
  "subject_types_supported": [
    "public"
  ]
}
```

Création et gestion d'un fournisseur OIDC (console)

Observez les instructions suivantes pour créer et gérer un fournisseur d'identité OIDC IAM dans la AWS Management Console.

> **Important**
>
> Si vous utilisez un fournisseur d'identité OIDC de Google, Facebook ou Amazon Cognito, ne créez pas de fournisseur d'identité IAM distinct selon cette procédure. Ces fournisseurs d'identité OIDC sont déjà intégrés AWS et sont disponibles pour votre usage. Au lieu de cela, créez de nouveaux rôles pour votre fournisseur d'identité, en consultant Création d'un rôle pour la fédération OpenID Connect (console).

Pour créer un fournisseur d'identité OIDC IAM (console)

1. Avant de créer un fournisseur d'identité OIDC IAM, vous devez enregistrer votre application auprès du fournisseur d'identité afin de recevoir un ID client. L'ID client (également appelé public ciblé) est un identifiant unique pour votre application. Il est émis lorsque vous enregistrez l'application auprès du fournisseur d'identité. Pour plus d'informations sur l'obtention d'un ID client, consultez la documentation de votre fournisseur d'identité.

   > **Note**
   >
   > AWS sécurise les communications avec certains fournisseurs d'identité OIDC (IdPs) via notre bibliothèque d'autorités de certification racine (CA) fiables au lieu d'utiliser une empreinte numérique de certificat pour vérifier le certificat de votre serveur IdP. Dans ce cas, votre empreinte numérique héritée est conservée dans la configuration, mais n'est plus utilisée pour la validation. Ces OIDC IdPs incluent Auth0,, GitHub GitLab, Google et ceux qui utilisent un compartiment Amazon S3 pour héberger un point de terminaison JSON Web Key Set (JWKS).

2. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
3. Dans le panneau de navigation, sélectionnez Fournisseurs d'identité, puis Ajouter un fournisseur.
4. Pour Configurer le fournisseur, sélectionnez OpenID Connect.
5. Sous URL du fournisseur, tapez l'URL du fournisseur d'identité. L'URL doit respecter les restrictions suivantes :

   - L'URL est sensible à la casse.
   - L'URL doit commencer par **`https://`**.
   - L'URL ne doit pas contenir de numéro de port.

- Au sein de votre Compte AWS, chaque fournisseur d'identité IAM OIDC doit utiliser une URL unique. Si vous essayez de soumettre une URL qui a déjà été utilisée pour un fournisseur OpenID Connect dans le Compte AWS, vous recevrez un message d'erreur.

6. Pour Audience, saisissez l'ID client de l'application que vous avez enregistrée auprès de l'IdP et dans Step 1 laquelle vous avez envoyé des demandes. AWS Si vous disposez d'ID client supplémentaires (également appelés publics ciblés) pour ce fournisseur d'identité, vous pouvez les ajouter ultérieurement sur la page de détails du fournisseur.

   > **Note**
   >
   > Si votre jeton IdP JWT inclut la `azp` réclamation, entrez cette valeur comme valeur d'audience.

7. (Facultatif) Pour Ajouter des balises, vous pouvez ajouter des paires clé-valeur pour vous aider à identifier et à organiser votre. IdPs Vous pouvez également utiliser des balises pour contrôler l'accès aux ressources AWS . Pour en savoir plus sur le balisage des fournisseurs d'identité OIDC IAM, reportez-vous à la section Balisage de fournisseurs d'identité OpenID Connect (OIDC). Choisissez Ajouter une balise. Saisissez des valeurs pour chaque paire clé-valeur de balise.
8. Vérifiez les informations que vous avez fournies. Lorsque vous avez terminé, sélectionnez Ajouter un fournisseur. IAM tentera de récupérer et d'utiliser l'empreinte numérique de l'autorité de certification intermédiaire supérieure du certificat du serveur IDP OIDC pour créer le fournisseur d'identité IAM OIDC.

   > **Note**
   >
   > La chaîne de certificats du fournisseur d'identité OIDC doit commencer par l'URL du domaine ou de l'émetteur, puis par le certificat intermédiaire et se terminer par le certificat racine. Si l'ordre de la chaîne de certificats est différent ou inclut des certificats dupliqués ou supplémentaires, vous recevez une erreur de non-concordance de signature et STS ne parvient pas à valider le jeton Web JSON (JWT). Corrigez l'ordre des certificats dans la chaîne renvoyée par le serveur pour résoudre l'erreur. Pour plus d'informations sur les normes relatives aux chaînes de certificats, consultez certificate_list dans la RFC 5246 sur le site Web de la série RFC.

9. Attribuez un rôle IAM à votre fournisseur d'identité pour autoriser les identités d'utilisateurs externes gérées par votre fournisseur d'identité à accéder aux AWS ressources de votre compte.

Pour en savoir plus sur la création de rôles pour la fédération d'identité, consultez la section Création d'un rôle pour un fournisseur d'identité tiers (fédération).

> **Note**
>
> L'OIDC IdPs utilisé dans une politique d'approbation des rôles doit se trouver dans le même compte que le rôle qui l'approuve.

Pour ajouter ou supprimer une empreinte pour un fournisseur d'identité OIDC IAM (console)

> **Note**
>
> AWS sécurise les communications avec certains fournisseurs d'identité OIDC (IdPs) via notre bibliothèque d'autorités de certification racine (CA) fiables au lieu d'utiliser une empreinte numérique de certificat pour vérifier le certificat de votre serveur IdP. Dans ce cas, votre empreinte numérique héritée est conservée dans la configuration, mais n'est plus utilisée pour la validation. Ces OIDC IdPs incluent Auth0,, GitHub GitLab, Google et ceux qui utilisent un compartiment Amazon S3 pour héberger un point de terminaison JSON Web Key Set (JWKS).

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Fournisseurs d'identité. Sélectionnez ensuite le nom du fournisseur d'identité IAM que vous souhaitez mettre à jour.
3. Dans la section Empreintes, sélectionnez Gérer. Pour saisir une nouvelle valeur d'empreinte, sélectionnez Ajouter une empreinte. Pour supprimer une empreinte numérique, sélectionnez Supprimer en regard de l'empreinte que vous souhaitez supprimer.

   > **Note**
   >
   > Un fournisseur d'identité OIDC IAM peut avoir entre 1 (minimum) et 5 empreintes.

   Lorsque vous avez terminé, sélectionnez Enregistrer les modifications.

Pour ajouter une audience pour un fournisseur d'identité OIDC IAM (console)

1. Dans le panneau de navigation, sélectionnez Identity providers (Fournisseurs d'identité), puis le nom du fournisseur d'identité IAM que vous voulez mettre à jour.
2. Dans la section Audiences, sélectionnez Actions et Ajouter une audience.
3. Entrez l'ID client de l'application que vous avez enregistrée auprès de l'IdP et dans Step 1 laquelle vous allez envoyer des demandes. AWS Sélectionnez ensuite Ajouter des audiences.

> **Note**
>
> Un fournisseur d'identité OIDC IAM peut avoir entre 1 (minimum) et 100 audiences (maximum).

Pour supprimer une audience pour un fournisseur d'identité OIDC IAM (console)

1. Dans le panneau de navigation, sélectionnez Identity providers (Fournisseurs d'identité), puis le nom du fournisseur d'identité IAM que vous voulez mettre à jour.
2. Dans la section Audiences, activez la case d'option en regard de l'audience que vous souhaitez supprimer, puis sélectionnez Actions.
3. Sélectionnez Supprimer l'audience. Une nouvelle fenêtre s'ouvre.
4. Si vous supprimez une audience, les identités fédérées avec l'audience ne peuvent pas endosser les rôles associés à l'audience. Dans la fenêtre, lisez le message d'avertissement et confirmez que vous souhaitez supprimer l'audience en saisissant le mot `remove` dans le champ.
5. Sélectionnez Supprimer pour supprimer l'audience.

Pour supprimer un fournisseur d'identité OIDC IAM (console)

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Fournisseurs d'identité.
3. Sélectionnez la case à cocher en regard du fournisseur d'identité IAM que vous souhaitez supprimer. Une nouvelle fenêtre s'ouvre.
4. Confirmez que vous souhaitez supprimer le fournisseur en saisissant le mot `delete` dans le champ. Ensuite, choisissez Supprimer.

Création et gestion d'un fournisseur d'identité OIDC IAM (AWS CLI)

Vous pouvez utiliser les AWS CLI commandes suivantes pour créer et gérer des fournisseurs d'identité IAM OIDC.

Pour créer un fournisseur d'identité OIDC IAM (AWS CLI)

1. (Facultatif) Pour obtenir la liste de tous les fournisseurs d'identité OIDC IAM de votre compte AWS , exécutez la commande suivante :
   - `aws iam list-open-id-connect-providers`
2. Pour créer un fournisseur d'identité OIDC IAM, exécutez la commande suivante :
   - `aws iam create-open-id-connect-provider`

Pour mettre à jour la liste des empreintes numériques de certificat de serveur pour un fournisseur d'identité OIDC IAM existant (AWS CLI)

- Pour mettre à jour la liste des empreintes de certificat de serveur pour un fournisseur d'identité OIDC IAM, exécutez la commande suivante :
  - `aws iam update-open-id-connect-provider-thumbprint`

Pour baliser un fournisseur d'identité OIDC IAM existant (AWS CLI)

- Pour baliser un fournisseur d'identité OIDC IAM existant, exécutez la commande suivante :
  - `aws iam tag-open-id-connect-provider`

Pour afficher la liste des balises d'un fournisseur d'identité OIDC IAM existant (AWS CLI)

- Pour afficher la liste des balises d'un fournisseur d'identité OIDC IAM existant, exécutez la commande suivante :
  - `aws iam list-open-id-connect-provider-tags`

Pour supprimer les balises d'un fournisseur d'identité OIDC IAM (AWS CLI)

- Pour supprimer les balises d'un fournisseur d'identité OIDC IAM existant, exécutez la commande suivante :
  - `aws iam untag-open-id-connect-provider`

Pour ajouter un ID client à un fournisseur d'identité OIDC IAM existant ou le supprimer (API AWS CLI)

1. (Facultatif) Pour obtenir la liste de tous les fournisseurs d'identité OIDC IAM de votre compte AWS , exécutez la commande suivante :
   - `aws iam list-open-id-connect-providers`
2. (Facultatif) Pour obtenir des informations détaillées sur un fournisseur d'identité OIDC IAM, exécutez la commande suivante :
   - `aws iam get-open-id-connect-provider`
3. Pour ajouter un nouvel ID client à un fournisseur d'identité OIDC IAM existant, exécutez la commande suivante :
   - `aws iam add-client-id-to-open-id-connect-provider`
4. Pour supprimer un client d'un fournisseur d'identité OIDC IAM existant, exécutez la commande suivante :
   - `aws iam remove-client-id-from-open-id-connect-provider`

Pour supprimer un fournisseur d'identité OIDC IAM (AWS CLI)

1. (Facultatif) Pour obtenir la liste de tous les fournisseurs d'identité OIDC IAM de votre compte AWS , exécutez la commande suivante :
   - `aws iam list-open-id-connect-providers`
2. (Facultatif) Pour obtenir des informations détaillées sur un fournisseur d'identité OIDC IAM, exécutez la commande suivante :
   - `aws iam get-open-id-connect-provider`
3. Pour supprimer un fournisseur d'identité OIDC IAM, exécutez la commande suivante :

- `aws iam delete-open-id-connect-provider`

Création et gestion d'un fournisseur d'identité OIDC (AWS API)

Vous pouvez utiliser les commandes de l'API IAM suivantes pour créer et gérer des fournisseurs OICD.

Pour créer un fournisseur d'identité (AWS API) IAM OIDC

1. (Facultatif) Pour obtenir la liste de tous les fournisseurs d'identité OIDC IAM de votre compte AWS , appelez l'opération suivante :
   - `ListOpenIDConnectProviders`
2. Pour créer un fournisseur d'identité OIDC IAM, appelez l'opération suivante :
   - `CreateOpenIDConnectProvider`

Pour mettre à jour la liste des empreintes numériques des certificats de serveur pour un fournisseur d'identité (API) IAM OIDC existant AWS

- Pour mettre à jour la liste des empreintes de certificat de serveur pour un fournisseur d'identité OIDC IAM, appelez l'opération suivante :
  - `UpdateOpenIDConnectProviderThumbprint`

Pour étiqueter un fournisseur d'identité (AWS API) IAM OIDC existant

- Pour baliser un fournisseur d'identité OIDC IAM existant, appelez l'opération suivante :
  - `TagOpenIDConnectProvider`

Pour répertorier les balises d'un fournisseur d'identité (AWS API) IAM OIDC existant

- Pour afficher la liste des balises d'un fournisseur d'identité OIDC IAM existant, appelez l'opération suivante :
  - `ListOpenIDConnectProviderTags`

Pour supprimer des balises sur un fournisseur d'identité (AWS API) IAM OIDC existant

- Pour supprimer les balises d'un fournisseur d'identité OIDC IAM existant, appelez l'opération suivante :
  - `UntagOpenIDConnectProvider`

Pour ajouter un ID client à un fournisseur d'identité OIDC IAM existant ou le supprimer (API AWS)

1. (Facultatif) Pour obtenir la liste de tous les fournisseurs d'identité OIDC IAM de votre compte AWS , appelez l'opération suivante :
   - `ListOpenIDConnectProviders`
2. (Facultatif) Pour obtenir des informations détaillées sur un fournisseur d'identité OIDC IAM, appelez l'opération suivante :
   - `GetOpenIDConnectProvider`
3. Pour ajouter un nouvel ID client à un fournisseur d'identité OIDC IAM existant, appelez l'opération suivante :
   - `AddClientIDToOpenIDConnectProvider`
4. Pour supprimer un ID client d'un fournisseur d'identité OIDC IAM existant, appelez l'opération suivante :
   - `RemoveClientIDFromOpenIDConnectProvider`

Pour supprimer un fournisseur d'identité (AWS API) IAM OIDC

1. (Facultatif) Pour obtenir la liste de tous les fournisseurs d'identité OIDC IAM de votre compte AWS , appelez l'opération suivante :
   - `ListOpenIDConnectProviders`
2. (Facultatif) Pour obtenir des informations détaillées sur un fournisseur d'identité OIDC IAM, appelez l'opération suivante :
   - `GetOpenIDConnectProvider`
3. Pour supprimer un fournisseur d'identité OIDC IAM, appelez l'opération suivante :

- `DeleteOpenIDConnectProvider`

Obtenir l'empreinte numérique d'un fournisseur d'identité OpenID Connect

Lorsque vous créez un fournisseur d'identité OpenID Connect (OIDC) dans IAM, IAM a besoin de l'empreinte numérique de l'autorité de certification (CA) intermédiaire supérieure qui a signé le certificat utilisé par le fournisseur d'identité externe (IdP). L'empreinte est une signature pour le certificat de l'autorité de certification qui a été utilisé pour émettre le certificat pour l'IdP compatible avec OIDC. Lorsque vous créez un fournisseur d'identité IAM OIDC, vous faites confiance aux identités authentifiées par cet IdP pour avoir accès à votre. Compte AWS En utilisant l'empreinte numérique du certificat de l'autorité de certification, vous faites confiance à tout certificat émis par cette autorité de certification avec le même nom DNS que celui enregistré. Ainsi, vous n'avez plus besoin de mettre à jour les approbations dans chaque compte lorsque vous renouvelez le certificat de signature de l'IdP.

> **Important**
>
> Dans la plupart des cas, le serveur de fédération utilise deux certificats différents.
>
> - Le premier établit une connexion HTTPS entre AWS et votre IdP. Cela doit être émis par une autorité de certification racine publique bien connue, telle que AWS Certificate Manager. Cela permet au client de vérifier la fiabilité et l'état du certificat.
> - Le second est utilisé pour crypter les jetons et doit être signé par un opérateur privé ou public racine CA.

Vous pouvez créer un fournisseur d'identité IAM OIDC à l'aide AWS Command Line Interface des outils pour Windows ou de l' PowerShellAPI IAM. Lorsque vous utilisez ces méthodes, vous avez la possibilité de fournir manuellement une empreinte numérique. Si vous choisissez de ne pas inclure d'empreinte numérique, IAM récupérera l'empreinte numérique intermédiaire supérieure du certificat du serveur IdP OIDC. Si vous choisissez d'inclure une empreinte numérique, vous devez l'obtenir manuellement et la fournir à. AWS

Lorsque vous créez un fournisseur d'identité OIDC avec la console IAM, IAM tente de récupérer pour vous l'empreinte numérique de l'autorité de certification intermédiaire supérieure du certificat du serveur IdP OIDC.

Nous vous recommandons également d'obtenir manuellement l'empreinte numérique de votre IdP OIDC et de vérifier qu'IAM a récupéré l'empreinte digitale correcte. Pour plus d'informations sur l'obtention des empreintes digitales des certificats, consultez les sections suivantes.

> **Note**
>
> AWS sécurise les communications avec certains fournisseurs d'identité OIDC (IdPs) via notre bibliothèque d'autorités de certification racine (CA) fiables au lieu d'utiliser une empreinte numérique de certificat pour vérifier le certificat de votre serveur IdP. Dans ce cas, votre empreinte numérique héritée est conservée dans la configuration, mais n'est plus utilisée pour la validation. Ces OIDC IdPs incluent Auth0,, GitHub GitLab, Google et ceux qui utilisent un compartiment Amazon S3 pour héberger un point de terminaison JSON Web Key Set (JWKS).

Pour résoudre les problèmes courants liés à la fédération IAM OIDC, consultez la section Résoudre les erreurs liées à OIDC sur Re:post. AWS

Obtenir l'empreinte numérique du certificat

Vous utilisez un navigateur Web et l'outil de ligne de commande OpenSSL pour obtenir l'empreinte numérique du certificat d'un fournisseur OIDC. Toutefois, il n'est pas nécessaire d'obtenir manuellement l'empreinte numérique du certificat pour créer un fournisseur d'identité IAM OIDC. Vous pouvez utiliser la procédure suivante pour obtenir l'empreinte numérique du certificat de votre fournisseur OIDC.

Pour obtenir l'empreinte d'un IdP OIDC

1. Avant d'obtenir l'empreinte pour un IdP OIDC, vous devez obtenir l'outil de ligne de commande OpenSSL. Cet outil vous permet de télécharger la chaîne de certificats de l'IdP OIDC et de fournir une empreinte du certificat final dans la chaîne de certificats. Si vous avez besoin d'installer et de configurer OpenSSL, suivez les instructions des sections Installer OpenSSL et Configurer OpenSSL.
2. Commencez par l'URL de l'IdP OIDC (par exemple, `https://server.example.com`), puis ajoutez `/.well-known/openid-configuration` pour former l'URL du document de configuration de l'IdP, comme suit :

   `https://server.example.com/.well-known/openid-configuration`

Ouvrez cette URL dans un navigateur web en remplaçant *serveur.exemple.com* par le nom du serveur du votre IdP.

3. Dans le document affiché, utilisez la fonction rechercher de votre navigateur web pour localiser le texte `"jwks_uri"`. Immédiatement après le texte `"jwks_uri"`, vous verrez apparaître deux points (:) suivis d'une URL. Copiez le nom de domaine complet de l'URL. N'incluez pas `https://` ou le chemin d'accès qui suit le domaine de niveau supérieur.

```
{
 "issuer": "https://accounts.example.com",
 "authorization_endpoint": "https://accounts.example.com/o/oauth2/v2/auth",
 "device_authorization_endpoint": "https://oauth2.exampleapis.com/device/code",
 "token_endpoint": "https://oauth2.exampleapis.com/token",
 "userinfo_endpoint": "https://openidconnect.exampleapis.com/v1/userinfo",
 "revocation_endpoint": "https://oauth2.exampleapis.com/revoke",
 "jwks_uri": "https://www.exampleapis.com/oauth2/v3/certs",
...
```

4. Utilisez l'outil de ligne de commande OpenSSL pour exécuter la commande suivante. Remplacez *clés.exemple.com* par le nom de domaine que vous avez obtenu dans Step 3.

```
openssl s_client -servername keys.example.com -showcerts -
connect keys.example.com:443
```

5. Dans votre fenêtre de commande, faites défiler jusqu'à afficher un certificat similaire à l'exemple suivant. Si plusieurs certificats sont affichés, recherchez le dernier certificat affiché (à la fin de la sortie de commande). Celui-ci contient le certificat de l'autorité de certification (CA) intermédiaire supérieure dans la chaîne d'autorité de certification.

```
-----BEGIN CERTIFICATE-----
 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
 21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
 rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
```

```
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
-----END CERTIFICATE-----
```

Copiez le certificat (notamment les lignes `-----BEGIN CERTIFICATE-----` et `-----END CERTIFICATE-----`) et collez-le dans un fichier texte. Puis enregistrez le fichier sous le nom **`certificate.crt`**.

> **Note**
>
> La chaîne de certificats du fournisseur d'identité OIDC doit commencer par l'URL du domaine ou de l'émetteur, puis par le certificat intermédiaire et se terminer par le certificat racine. Si l'ordre de la chaîne de certificats est différent ou inclut des certificats dupliqués ou supplémentaires, vous recevez une erreur de non-concordance de signature et STS ne parvient pas à valider le jeton Web JSON (JWT). Corrigez l'ordre des certificats dans la chaîne renvoyée par le serveur pour résoudre l'erreur. Pour plus d'informations sur les normes relatives aux chaînes de certificats, consultez certificate_list dans la RFC 5246 sur le site Web de la série RFC.

6. Utilisez l'outil de ligne de commande OpenSSL pour exécuter la commande suivante.

```
openssl x509 -in certificate.crt -fingerprint -sha1 -noout
```

Votre fenêtre de commande affiche l'empreinte du certificat qui semble similaire à l'exemple suivant :

```
SHA1 Fingerprint=99:0F:41:93:97:2F:2B:EC:F1:2D:DE:DA:52:37:F9:C9:52:F2:0D:9E
```

Supprimez les deux points (:) de cette chaîne pour générer l'empreinte finale, comme ceci :

```
990F4193972F2BECF12DDEDA5237F9C952F20D9E
```

7. Si vous créez le fournisseur d'identité IAM OIDC à l'aide de l' AWS CLI API Tools for Windows ou IAM PowerShell, la fourniture d'une empreinte numérique est facultative. Si vous choisissez de ne pas inclure d'empreinte numérique lors de la création, IAM récupérera l'empreinte numérique

intermédiaire supérieure du certificat du serveur IdP OIDC. Une fois le fournisseur d'identité IAM OIDC créé, vous pouvez comparer cette empreinte numérique à celle récupérée par IAM.

Si vous créez le fournisseur d'identité IAM OIDC dans la console IAM, celle-ci tente de récupérer pour vous l'empreinte numérique de l'autorité de certification intermédiaire supérieure du certificat du serveur IdP OIDC. Vous pouvez comparer cette empreinte numérique à celle récupérée par IAM. Une fois le fournisseur d'identité IAM OIDC créé, vous pouvez consulter l'empreinte numérique du fournisseur d'identité IAM OIDC dans l'onglet Endpoint verification de la page de la console de synthèse du fournisseur OIDC.

> **Important**
>
> Si l'empreinte numérique que vous avez obtenue ne correspond pas à celle que vous voyez dans les détails de l'empreinte numérique du fournisseur d'identité IAM OIDC, vous ne devez pas utiliser le fournisseur OIDC. Vous devez plutôt supprimer le fournisseur OIDC créé, puis réessayer de le créer après un certain temps. Vérifiez que les empreintes du pouce correspondent avant d'utiliser le fournisseur. Si les empreintes ne correspondent toujours pas après une seconde tentative, visitez le Forum IAM pour contacter AWS.

Installer OpenSSL

Si OpenSSL n'est pas déjà installée, suivez les instructions de cette section.

Pour installer OpenSSL sous Linux ou Unix

1. Accédez à OpenSSL: Source, Tarballs (https://openssl.org/source/).
2. Téléchargez la source la plus récente et créez le package.

Pour installer OpenSSL sous Windows

1. Accédez à OpenSSL: Binary Distributions (https://wiki.openssl.org/index.php/Binaries) pour obtenir une liste des sites à partir desquels installer la version Windows.
2. Suivez les instructions sur le site que vous avez sélectionné pour démarrer l'installation.
3. Si vous êtes invité à installer le package Microsoft Visual C++ 2008 Redistributable et qu'il n'est pas déjà installé sur votre système, choisissez le lien de téléchargement correspondant à votre

environnement. Suivez les instructions fournies par l'Assistant d'installation de Microsoft Visual C ++ 2008 Redistributable.

> **Note**
>
> Si vous n'êtes pas sûr que Microsoft Visual C++ 2008 Redistributable soit déjà installé sur votre système, vous pouvez essayer d'installer d'abord OpenSSL. Le programme d'installation d'OpenSSL affiche une alerte si Microsoft Visual C++ 2008 Redistributable n'est pas encore installé. Veillez à installer l'architecture (32 ou 64 bits) correspondant à la version d'OpenSSL que vous installez.

4. Après avoir installé le package Microsoft Visual C++ 2008 Redistributable, sélectionnez la version des fichiers binaires OpenSSL correspondant à votre environnement et enregistrez le fichier localement. Démarrer l'Assistant d'installation d'OpenSSL.
5. Suivez les instructions décrites dans l'Assistant d'installation d'OpenSSL.

Configurer OpenSSL

Avant d'utiliser les commandes OpenSSL, vous devez configurer le système d'exploitation afin qu'il sache où OpenSSL est installé.

Pour configurer OpenSSL sous Linux ou Unix

1. Sur la ligne de commande, définissez la variable `OpenSSL_HOME` à l'emplacement d'installation d'OpenSSL :

```
$ export OpenSSL_HOME=path_to_your_OpenSSL_installation
```

2. Définissez le chemin d'accès de sorte à inclure l'installation d'OpenSSL :

```
$ export PATH=$PATH:$OpenSSL_HOME/bin
```

> **Note**
>
> Les modifications apportées aux variables d'environnement à l'aide de la commande `export` sont valides uniquement pour la session actuelle. Vous pouvez apporter des modifications persistantes aux variables d'environnement en les définissant dans votre

fichier de configuration de shell. Pour plus d'informations, consultez la documentation de votre système d'exploitation.

Pour configurer OpenSSL sous Windows

1. Ouvrez une fenêtre d'invite de commande.
2. Définissez la variable `OpenSSL_HOME` à l'emplacement d'installation d'OpenSSL :

```
C:\> set OpenSSL_HOME=path_to_your_OpenSSL_installation
```

3. Définissez la variable `OpenSSL_CONF` à l'emplacement du fichier de configuration dans votre installation d'OpenSSL :

```
C:\> set OpenSSL_CONF=path_to_your_OpenSSL_installation\bin\openssl.cfg
```

4. Définissez le chemin d'accès de sorte à inclure l'installation d'OpenSSL :

```
C:\> set Path=%Path%;%OpenSSL_HOME%\bin
```

> **Note**
>
> Toutes les modifications que vous apportez aux variables d'environnement Windows dans une fenêtre d'invite de commandes ne sont valides que pour la session de ligne de commande en cours. Vous pouvez apporter des modifications persistantes aux variables d'environnement en les définissant comme propriétés système. Les procédures précises dépendent de la version de Windows que vous utilisez. (Par exemple, dans Windows 7, ouvrez le Panneau de configuration, puis Système et sécurité, Système. Ensuite, choisissez Paramètres système avancés, Avancés, Variables d'environnement.) Pour de plus amples informations, veuillez consulter la documentation Windows.

Ressources supplémentaires pour la fédération OIDC

Les ressources suivantes peuvent vous aider à en savoir plus sur la fédération OIDC :

- Utilisez OpenID Connect dans vos GitHub flux de travail en configurant OpenID Connect dans Amazon Web Services

- Amazon Cognito Identity dans le Guide Amplify Libraries for Android et Amazon Cognito Identity dans le Guide Amplify Libraries for Swift.
- Le blog Automatisation des rôles d'identité Web AWS IAM basés sur OpenID Connect à l'aide de Microsoft Entra ID on AWS the Partner Network (APN) explique comment authentifier des processus automatisés en arrière-plan ou des applications exécutées en dehors de l'autorisation OIDC. AWS machine-to-machine
- L'article Fédération d'identité Web avec applications mobiles traite de la fédération OIDC et montre un exemple d'utilisation de la fédération OIDC pour accéder au contenu d'Amazon S3.

Fédération SAML 2.0

AWS prend en charge la fédération d'identité avec SAML 2.0 (Security Assertion Markup Language 2.0), un standard ouvert utilisé par de nombreux fournisseurs d'identité (IdPs). Cette fonctionnalité permet l'authentification unique fédérée (SSO), afin que les utilisateurs puissent se connecter aux opérations de l' AWS API AWS Management Console ou les appeler sans que vous ayez à créer un utilisateur IAM pour tous les membres de votre organisation. En utilisant SAML, vous pouvez simplifier le processus de configuration de la fédération avec AWS, car vous pouvez utiliser le service de l'IdP au lieu d'écrire un code proxy d'identité personnalisé.

La fédération IAM prend en charge les cas d'utilisation suivants :

- Accès fédéré pour permettre à un utilisateur ou à une application de votre organisation d'appeler des opérations AWS d'API. Ce cas d'utilisation est décrit dans la section suivante. Vous utilisez une assertion SAML (dans le cadre de la réponse d'authentification) qui est générée dans votre organisation pour l'obtention d'informations d'identification temporaires. Ce scénario est similaire à d'autres scénarios de fédération pris en charge par IAM, tels que ceux décrits dans Demande d'informations d'identification temporaires de sécurité et Fédération OIDC. Toutefois, les systèmes SAML 2.0 de votre IdPs organisation gèrent de nombreux détails lors de l'exécution de l'authentification et de la vérification des autorisations.
- Authentification unique (SSO) basée sur le Web AWS Management Console auprès de votre organisation. Les utilisateurs peuvent se AWS connecter à un portail de votre organisation hébergé par un IdP compatible SAML 2.0, sélectionner une option d'accès et être redirigés vers la console sans avoir à fournir d'informations de connexion supplémentaires. Vous pouvez utiliser un IdP SAML tiers pour établir un accès SSO à la console ou créer un IdP personnalisé pour autoriser l'accès de vos utilisateurs externes à la console. Pour plus d'informations sur la création d'un IdP

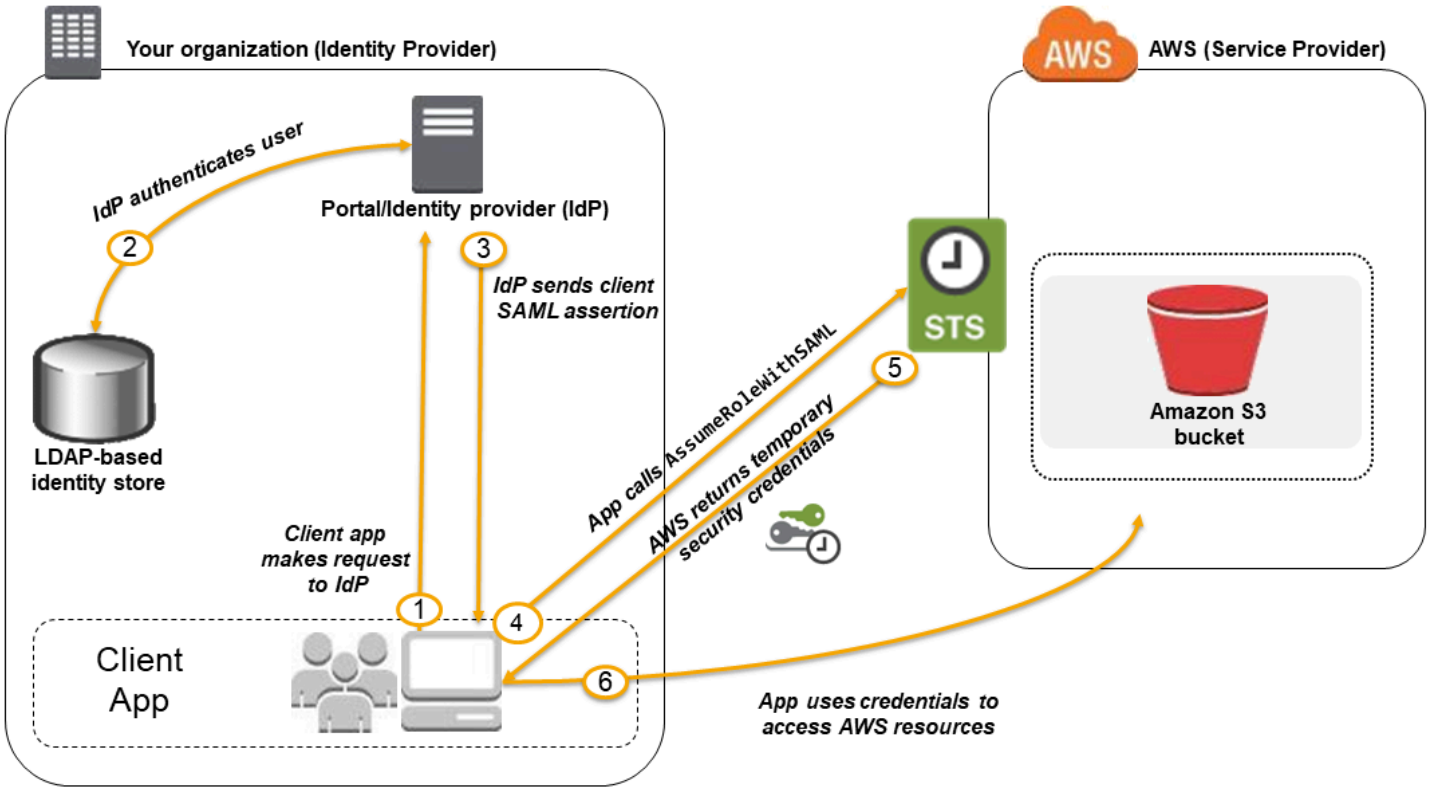
personnalisé, consultez la page Activation de l'accès à la AWS console par un courtier d'identité personnalisé.

Rubriques

- Utilisation de la fédération SAML pour l'accès à l'API AWS
- Présentation de la configuration de la fédération SAML 2.0
- Vue d'ensemble du rôle permettant d'autoriser l'accès fédéré par SAML à vos ressources AWS
- Identification unique des utilisateurs dans la fédération SAML
- Création d'un fournisseur d'identité SAML dans IAM
- Configurez votre IdP SAML 2.0 en vous fiant à la confiance des parties et en ajoutant des réclamations
- Intégrez des fournisseurs de solutions SAML tiers avec AWS
- Configurer les assertions SAML pour la réponse d'authentification
- Permettre aux utilisateurs fédérés SAML 2.0 d'accéder au AWS Management Console

Utilisation de la fédération SAML pour l'accès à l'API AWS

Supposons que vous voulez permettre aux employés de copier les données de leurs ordinateurs dans un dossier de sauvegarde. Vous créez une application que les utilisateurs peuvent exécuter sur leurs ordinateurs. Sur le back-end, l'application lit et écrit des objets dans un compartiment Amazon S3. Les utilisateurs n'ont pas un accès direct à AWS. À la place, le processus suivant est utilisé :



1. A l'aide d'une application cliente, un utilisateur de l'organisation demande son authentification par l'IdP de l'organisation.
2. L'IdP authentifie l'utilisateur en le comparant à la base d'identités de l'organisation.
3. L'IdP crée une assertion SAML à l'aide des informations concernant l'utilisateur et l'envoie à l'application client.
4. L'application cliente appelle l' AWS STS `AssumeRoleWithSAML`API en transmettant l'ARN du fournisseur SAML, l'ARN du rôle à assumer et l'assertion SAML de l'IdP.
5. La réponse de l'API à l'application cliente inclut des informations d'identification de sécurité temporaires.
6. L'application cliente utilise ces informations d'identification de sécurité temporaires pour appeler les opérations d'API Amazon S3.

Présentation de la configuration de la fédération SAML 2.0

Avant de pouvoir utiliser la fédération basée sur SAML 2.0 comme décrit dans le scénario et le schéma précédents, vous devez configurer l'IdP de votre organisation et le vôtre de manière à ce qu'ils se fassent mutuellement confiance. Compte AWS La procédure suivante décrit le processus général permettant de configurer cette relation d'approbation. Votre organisation doit utiliser un IdP

prenant en charge SAML 2.0, comme Microsoft Active Directory Federation Service (services ADFS qui font partie de Windows Server), Shibboleth, ou tout autre fournisseur compatible avec SAML 2.0.

> **Note**
>
> Pour améliorer la résilience de la fédération, nous vous recommandons de configurer votre IdP et votre fédération AWS pour prendre en charge plusieurs points de terminaison de connexion SAML. Pour plus de détails, consultez l'article du blog sur la AWS sécurité Comment utiliser les points de terminaison SAML régionaux pour le basculement.

Configurez l'IdP de votre organisation et AWS faites-vous confiance

1. Inscrivez-vous AWS en tant que fournisseur de services (SP) auprès de l'IdP de votre organisation. Utilisez le document de métadonnées SAML à partir de `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`

   Pour obtenir la liste des valeurs possibles de *region-code*, consultez la colonne Region (Région) des AWS Sign-In endpoints (Points de terminaison de connexion).

   Vous pouvez éventuellement utiliser le document de métadonnées SAML à partir de `https://signin.aws.amazon.com/static/saml-metadata.xml` .
2. À l'aide de l'IdP de l'organisation, vous générez un fichier XML de métadonnées équivalent, capable de décrire cet IdP en tant que fournisseur d'identité IAM dans AWS. Il doit inclure le nom de l'émetteur, une date de création, une date d'expiration et les clés qui AWS peuvent être utilisées pour valider les réponses d'authentification (assertions) de votre organisation.
3. Dans la console IAM, vous créez un fournisseur d'identité SAML. Dans le cadre de ce processus, vous téléchargez le document de métadonnées SAML qui ont été produits par l'IdP de votre organisation dans. Step 2 Pour plus d’informations, consultez Création d'un fournisseur d'identité SAML dans IAM.
4. Dans IAM, créez un ou plusieurs rôles IAM. Dans la politique de confiance du rôle, vous définissez le fournisseur SAML comme principal, ce qui établit une relation de confiance entre votre organisation et AWS. La politique d'autorisation du rôle détermine les actions que les utilisateurs de l'organisation sont autorisés à effectuer dans AWS. Pour plus d’informations, consultez Création d'un rôle pour un fournisseur d'identité tiers (fédération).

> **Note**
>
> Les IdP SAML utilisés dans une politique d'approbation de rôle doivent se trouver dans le même compte que le rôle.

5. Dans l'IdP de l'organisation, vous définissez les assertions qui associent les utilisateurs et les groupes de l'organisation aux rôles IAM. Notez que différents utilisateurs et groupes de l'organisation peuvent être associés à différents rôles IAM. La procédure exacte pour leur mappage dépend de l'IdP utilisé. Dans le scénario précédent d'un dossier Amazon S3 pour les utilisateurs, tous les utilisateurs peuvent être associés au rôle qui fournit les autorisations Amazon S3. Pour plus d'informations, consultez Configurer les assertions SAML pour la réponse d'authentification.

   Si votre IdP active l'authentification unique sur la AWS console, vous pouvez configurer la durée maximale des sessions de console. Pour plus d'informations, consultez Permettre aux utilisateurs fédérés SAML 2.0 d'accéder au AWS Management Console.

6. Dans l'application que vous créez, vous appelez l' AWS Security Token Service `AssumeRoleWithSAML`API en lui transmettant l'ARN du fournisseur SAML dans lequel vous l'avez créé Step 3, l'ARN du rôle dans lequel vous devez supposer que vous l'avez créé et l'assertion SAML concernant l'utilisateur actuel que vous obtenez de votre IdP. Step 4 AWS s'assure que la demande pour assumer le rôle provient de l'IdP référencé dans le fournisseur SAML.

   Pour plus d'informations, consultez AssumeRoleWithSAML dans la référence de l'AWS Security Token Service API.

7. Si la demande aboutit, l'API retourne des informations d'identification de sécurité temporaires que votre application peut utiliser pour adresser des demandes signées à AWS. Votre application dispose d'informations sur l'utilisateur actuel et peut accéder aux compartiments Amazon S3 spécifiques à celui-ci, comme décrit dans le scénario précédent.

Vue d'ensemble du rôle permettant d'autoriser l'accès fédéré par SAML à vos ressources AWS

Les rôles que vous créez dans IAM définissent les actions que les utilisateurs fédérés de votre organisation sont en mesure d'effectuer dans AWS. Lorsque vous créez la politique d'approbation pour le rôle, vous spécifiez le fournisseur d'identité SAML créé précédemment en tant que

`Principal`. Vous pouvez également ajouter une `Condition` à la politique d'approbation, afin d'autoriser uniquement les utilisateurs correspondant à certains attributs SAML à accéder au rôle. Par exemple, il est possible de spécifier que seuls les utilisateurs dont l'affiliation SAML est `staff` (comme stipulé sur https://openidp.feide.no) sont autorisés à accéder au rôle, comme illustré dans l'exemple de politique suivant :

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/
ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {
      "StringEquals": {
        "saml:aud": "https://signin.aws.amazon.com/saml",
        "saml:iss": "https://openidp.feide.no"
      },
      "ForAllValues:StringLike": {"saml:edupersonaffiliation": ["staff"]}
    }
  }]
}
```

> **Note**
>
> Les IdP SAML utilisés dans une politique d'approbation de rôle doivent se trouver dans le même compte que le rôle.

Pour plus d'informations sur les clés SAML que vous pouvez utiliser dans une politique, consultez Clés disponibles pour la fédération SAML AWS STS.

Vous pouvez inclure des points de terminaison régionaux pour l'attribut `saml:aud` à l'adresse `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`. Pour obtenir la liste des valeurs possibles de *`region-code`*, consultez la colonne Region (Région) des AWS Sign-In endpoints (Points de terminaison de connexion).

Vous spécifiez la politique d'autorisation dans le rôle comme pour tout autre rôle. Par exemple, si les utilisateurs de votre organisation sont autorisés à administrer des instances Amazon Elastic Compute

Cloud, vous devez explicitement autoriser les actions Amazon EC2 dans la politique d'autorisation, telles que celles figurant dans la politique gérée d'Amazon FullAccess EC2.

Identification unique des utilisateurs dans la fédération SAML

Lors de la création de politiques d'accès dans IAM, il est souvent utile de pouvoir spécifier des autorisations en fonction de l'identité des utilisateurs. Par exemple, dans le cas d'utilisateurs fédérés à l'aide de SAML, une application peut conserver les informations dans Amazon S3 à l'aide d'une structure similaire à celle-ci :

```
myBucket/app1/user1
myBucket/app1/user2
myBucket/app1/user3
```

Vous pouvez créer le compartiment (`myBucket`) et le dossier (`app1`) via la console Amazon S3 ou le AWS CLI, car il s'agit de valeurs statiques. Toutefois, les dossiers spécifiques aux utilisateurs (*`utilisateur1`*, *`utilisateur2`*, *`utilisateur3`*, etc.) doivent être créés à l'aide de code lors de l'exécution, car la valeur qui identifie l'utilisateur n'est pas connue jusqu'à la première authentification de l'utilisateur via le processus de fédération.

Pour créer des politiques qui référencent des détails spécifiques à l'utilisateur dans le nom d'une ressource, l'identité de l'utilisateur doit figurer dans des clés SAML pouvant être utilisées dans les conditions des politiques. Les clés suivantes sont disponibles pour la fédération basée sur SAML 2.0 à utiliser dans les politiques IAM. Vous pouvez utiliser les valeurs retournées par les clés suivantes pour créer des identifiants utilisateur uniques pour des ressources comme des dossiers Amazon S3.

- `AWS`. Valeur de hachage basée sur la concaténation de la valeur de réponse `Issuer` (`saml:iss`) et d'une chaîne avec l'ID de compte `saml:namequalifier` et le nom convivial (dernière partie de l'ARN) du fournisseur SAML dans IAM. La concaténation de l'ID de compte et du nom convivial du fournisseur SAML est disponible dans les politiques IAM en tant que clé `saml:doc`. L'ID de compte et le nom du fournisseur doivent être séparés par un caractère « / », comme dans « 123456789012/provider_name ». Pour plus d'informations, reportez-vous à la clé `saml:doc` dans Clés disponibles pour la fédération SAML AWS STS.

  La combinaison de `NameQualifier` et `Subject` permet d'identifier de manière unique un utilisateur fédéré. L'exemple de pseudo-code suivant montre comment cette valeur est calculée. Dans ce pseudo-code, + indique une concaténation, `SHA1` représente une fonction qui génère un résumé de message à l'aide de SHA-1 et `Base64` représente une fonction qui génère une version encodée en Base64 de la sortie de hachage.

```
Base64 ( SHA1 ( "https://example.com/saml" + "123456789012" + "/
MySAMLIdP" ) )
```

Pour plus d'informations sur les clés de politique disponibles pour la fédération SAML, consultez Clés disponibles pour la fédération SAML AWS STS.

- `saml:sub` (chaîne). Objet de la demande, qui inclut une valeur qui identifie de manière unique un utilisateur individuel au sein d'une organisation (par exemple, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).
- `saml:sub_type` (chaîne). Cette clé peut être `persistent`, `transient` ou l'URI `Format` complet des éléments `Subject` et `NameID` utilisés dans votre assertion SAML. Une va leur `persistent` indique que la valeur de `saml:sub` reste la même pour l'utilisateur pour toutes les sessions. Si la valeur est `transient`, l'utilisateur a une valeur `saml:sub` différente pour chaque session. Pour plus d'informations sur l'attribut `NameID` de l'élément `Format`, consultez Configurer les assertions SAML pour la réponse d'authentification.

L'exemple suivant illustre une politique d'autorisation qui utilise les clés précédentes pour accorder des autorisations à un dossier spécifique à un utilisateur dans Amazon S3. La politique suppose que les objets Amazon S3 sont identifiés à l'aide d'un préfixe qui inclut à la fois `saml:namequalifier` et `saml:sub`. Notez que l'élément `Condition` inclut un test pour vérifier que la valeur de `saml:sub_type` est définie sur `persistent`. Si la valeur est `transient`, l'élément `saml:sub` de l'utilisateur peut être différent pour chaque session et vous ne devez pas combiner les valeurs pour identifier des dossiers spécifiques aux utilisateurs.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": [
      "arn:aws:s3:::exampleorgBucket/backup/${saml:namequalifier}/${saml:sub}",
      "arn:aws:s3:::exampleorgBucket/backup/${saml:namequalifier}/${saml:sub}/*"
    ],
    "Condition": {"StringEquals": {"saml:sub_type": "persistent"}}
  }
```

```
}
```

Pour plus d'informations sur le mappage d'assertions de l'IdP et les clés de politique, consultez Configurer les assertions SAML pour la réponse d'authentification.

Création d'un fournisseur d'identité SAML dans IAM

Un fournisseur d'identité SAML 2.0 IAM est une entité dans IAM qui décrit un service de fournisseur d'identité (IdP) externe prenant en charge la norme SAML 2.0 (Security Assertion Markup Language 2.0). Vous utilisez un fournisseur d'identité IAM lorsque vous souhaitez établir un lien de confiance entre un IdP compatible SAML tel que Shibboleth ou Active Directory Federation Services AWS et afin que les utilisateurs de votre organisation puissent accéder aux ressources. AWS Les fournisseurs d'identité SAML IAM sont utilisés en tant que principaux dans une politique d'approbation IAM.

Pour plus d'informations sur ce scénario, consultez Fédération SAML 2.0.

Vous pouvez créer et gérer un fournisseur d'identité IAM dans AWS Management Console ou avec des AWS CLI outils pour Windows ou PowerShell des appels AWS d'API.

Après avoir créé un fournisseur SAML, vous devez créer un ou plusieurs rôles IAM. Un rôle est une identité AWS qui ne possède pas ses propres informations d'identification (comme le fait un utilisateur). Mais dans ce contexte, un rôle est attribué dynamiquement à un utilisateur fédéré qui est authentifié par le fournisseur d'identité (IdP) de votre organisation. Le rôle permet à l'IdP de votre organisation de demander des informations d'identification de sécurité temporaires pour accéder à AWS. Les politiques attribuées au rôle déterminent ce que les utilisateurs fédérés sont autorisés à faire dans AWS ce rôle. Pour créer un rôle pour la fédération SAML, consultez Création d'un rôle pour un fournisseur d'identité tiers (fédération).

Enfin, après avoir créé le rôle, vous complétez l'approbation SAML en configurant votre IdP avec les AWS informations et les rôles que vous souhaitez que vos utilisateurs fédérés utilisent. Il s'agit de la configuration de la relation d'approbation des parties utilisatrices entre votre IdP et AWS. Pour configurer la relation d'approbation des parties utilisatrices, consultez Configurez votre IdP SAML 2.0 en vous fiant à la confiance des parties et en ajoutant des réclamations.

Rubriques

- Prérequis
- Création et gestion d'un fournisseur d'identité IAM SAML (console)
- Création et gestion d'un fournisseur d'identité IAM SAML ()AWS CLI

- Création et gestion d'un fournisseur d'identité (AWS API) IAM SAML

Prérequis

Avant de créer un fournisseur d'identité SAML, vous devez disposer des informations suivantes auprès de votre IdP.

- Obtenez le document de métadonnées SAML auprès de votre IdP. Ce document inclut le nom de l'auteur, des informations d'expiration et des clés qui peuvent être utilisées pour valider les réponses d'authentification (assertions) SAML reçues du fournisseur d'identité (IdP). Pour générer le document de métadonnées, utilisez le logiciel de gestion des identités fourni par votre IdP externe.

  > **Important**
  >
  > Ce fichier de métadonnées inclut le nom de l'auteur, des informations d'expiration et des clés qui peuvent être utilisées pour valider les réponses d'authentification (assertions) SAML reçues du fournisseur d'identité (IdP). Le fichier de métadonnées doit être codé au format UTF-8 sans marque d'ordre d'octet (BOM). Pour supprimer la marque d'ordre d'octet (BOM), vous pouvez coder le fichier au format UTF-8 à l'aide d'un outil d'édition de texte, tel que Notepad++.
  >
  > Le certificat x.509 inclus comme partie du document des métadonnées SAML doit utiliser une taille de clé au moins égale à 1 024 bits. De plus, le certificat x.509 doit également être exempt de toute extension répétée. Vous pouvez utiliser des extensions, mais elles ne peuvent apparaître qu'une seule fois dans le certificat. Si le certificat x.509 ne répond à aucune des conditions, la création de l'IdP échoue et renvoie une erreur « impossible d'analyser les métadonnées ».
  >
  > Comme défini par la version 1.0 du profil d'interopérabilité des métadonnées SAML V2.0, IAM n'évalue ni ne prend aucune mesure concernant l'expiration du certificat X.509 du document de métadonnées.

Pour obtenir des instructions sur la façon de configurer la plupart des éléments disponibles IdPs pour les utiliser AWS, notamment sur la façon de générer le document de métadonnées SAML requis, consultezIntégrez des fournisseurs de solutions SAML tiers avec AWS.

Pour obtenir de l'aide concernant la fédération SAML, consultez la section Résolution des problèmes liés à la fédération SAML.

Création et gestion d'un fournisseur d'identité IAM SAML (console)

Vous pouvez utiliser le AWS Management Console pour créer, mettre à jour et supprimer des fournisseurs d'identité IAM SAML. Pour obtenir de l'aide concernant la fédération SAML, consultez la section Résolution des problèmes liés à la fédération SAML.

Pour créer un fournisseur d'identité SAML IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Fournisseurs d'identité, puis Ajouter un fournisseur.
3. Pour Configurer le fournisseur, sélectionnez SAML.
4. Saisissez un nom pour le fournisseur d'identité.
5. Pour Document de métadonnées, sélectionnez Choisir un fichier, spécifiez le document de métadonnées SAML que vous avez téléchargé dans the section called “Prérequis”.
6. (Facultatif) Pour Ajouter des balises, vous pouvez ajouter des paires clé-valeur pour vous aider à identifier et à organiser votre. IdPs Vous pouvez également utiliser des balises pour contrôler l'accès aux ressources AWS . Pour en savoir plus sur le balisage des fournisseurs d'identité SAML, reportez-vous à la section Balisage de fournisseurs d'identité SAML IAM.

   Choisissez Ajouter une balise. Saisissez des valeurs pour chaque paire clé-valeur de balise.
7. Vérifiez les informations que vous avez fournies. Lorsque vous avez terminé, sélectionnez Ajouter un fournisseur.
8. Attribuez un rôle IAM à votre fournisseur d'identité pour autoriser les identités d'utilisateurs externes gérées par votre fournisseur d'identité à accéder aux AWS ressources de votre compte. Pour en savoir plus sur la création de rôles pour la fédération d'identité, consultez la section Création d'un rôle pour un fournisseur d'identité tiers (fédération).

> **Note**
>
> Les IdP SAML utilisés dans une politique d'approbation de rôle doivent se trouver dans le même compte que le rôle.

Pour supprimer un fournisseur SAML (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Fournisseurs d'identité.
3. Activez la case d'option en regard du fournisseur d'identité que vous souhaitez supprimer.
4. Sélectionnez Delete (Supprimer). Une nouvelle fenêtre s'ouvre.
5. Confirmez que vous souhaitez supprimer le fournisseur en saisissant le mot `delete` dans le champ. Ensuite, choisissez Supprimer.

Création et gestion d'un fournisseur d'identité IAM SAML ()AWS CLI

Vous pouvez utiliser le AWS CLI pour créer, mettre à jour et supprimer des fournisseurs SAML. Pour obtenir de l'aide concernant la fédération SAML, consultez la section Résolution des problèmes liés à la fédération SAML.

Pour créer un fournisseur d'identité IAM et télécharger un document de métadonnées (AWS CLI)

- Exécutez cette commande : `aws iam create-saml-provider`

Pour mettre à jour un fournisseur d'identité IAM SAML ()AWS CLI

- Exécutez cette commande : `aws iam update-saml-provider`

Pour baliser un fournisseur d'identité IAM existant (AWS CLI)

- Exécutez cette commande : `aws iam tag-saml-provider`

Pour afficher la liste des balises d'un fournisseur d'identité IAM existant (AWS CLI)

- Exécutez cette commande : `aws iam list-saml-provider-tags`

Pour supprimer les balises d'un fournisseur d'identité IAM (AWS CLI) existant

- Exécutez cette commande : `aws iam untag-saml-provider`

Pour supprimer un fournisseur d'identité SAML IAM (AWS CLI)

1. (Facultatif) Pour répertorier des informations pour tous les fournisseurs, comme l'ARN, la date de création et l'expiration, exécutez la commande suivante :

   - `aws iam list-saml-providers`
2. (Facultatif) Pour obtenir des informations sur un fournisseur spécifique, telles que l'ARN, la date de création, la date d'expiration, les paramètres de chiffrement et les informations de clé privée, exécutez la commande suivante :

   - `aws iam get-saml-provider`
3. Pour supprimer un fournisseur d'identité IAM, exécutez la commande suivante :

   - `aws iam delete-saml-provider`

Création et gestion d'un fournisseur d'identité (AWS API) IAM SAML

Vous pouvez utiliser l' AWS API pour créer, mettre à jour et supprimer des fournisseurs SAML. Pour obtenir de l'aide concernant la fédération SAML, consultez la section Résolution des problèmes liés à la fédération SAML.

Pour créer un fournisseur d'identité IAM et télécharger un document de métadonnées (AWS API)

- Appelez cette opération : `CreateSAMLProvider`

Pour mettre à jour un fournisseur d'identité (AWS API) IAM SAML

- Appelez cette opération : `UpdateSAMLProvider`

Pour baliser un fournisseur d'identité (AWS API) IAM existant

- Appelez cette opération : `TagSAMLProvider`

Pour répertorier les balises d'un fournisseur d'identité (AWS API) IAM existant

- Appelez cette opération : `ListSAMLProviderTags`

Pour supprimer des balises sur un fournisseur d'identité (AWS API) IAM existant

- Appelez cette opération : `UntagSAMLProvider`

Pour supprimer un fournisseur d'identité (AWS API) IAM

1. (Facultatif) Pour répertorier les informations relatives à tous IdPs, telles que l'ARN, la date de création et l'expiration, effectuez l'opération suivante :
   - `ListSAMLProviders`
2. (Facultatif) Pour obtenir des informations sur un fournisseur spécifique, telles que l'ARN, la date de création, la date d'expiration, les paramètres de chiffrement et les informations de clé privée, effectuez l'opération suivante :
   - `GetSAMLProvider`
3. Pour supprimer un IdP, appelez l'opération suivante :
   - `DeleteSAMLProvider`

Configurez votre IdP SAML 2.0 en vous fiant à la confiance des parties et en ajoutant des réclamations

Lorsque vous créez un fournisseur d'identité IAM et le rôle pour l'accès SAML, vous donnez des informations à AWS sur le fournisseur d'identité (IdP) externe et sur les actions que les utilisateurs sont autorisés à effectuer. L'étape suivante consiste à en informer l'IdP en AWS tant que fournisseur de services. Il s'agit là de l'ajout d'une relation d'approbation des parties utilisatrices entre votre IdP et AWS. Le processus exact utilisé pour ajouter une approbation de partie utilisatrice dépend de l'IdP que vous utilisez. Pour plus d'informations, consultez la documentation de votre logiciel de gestion des identités.

Beaucoup vous IdPs permettent de spécifier une URL à partir de laquelle l'IdP peut lire un document XML contenant des informations et des certificats relatifs à une partie utilisatrice. Pour AWS, utilisez `https://`*`region-code`*`.signin.aws.amazon.com/static/saml-metadata.xml` ou`https://signin.aws.amazon.com/static/saml-metadata.xml`. Pour obtenir la liste des valeurs possibles de *`region-code`*, consultez la colonne Region (Région) des AWS Sign-In endpoints (Points de terminaison de connexion).

S'il n'est pas possible de spécifier directement une URL, téléchargez le document XML à partir de l'URL ci-dessus et importez-le dans l'application de votre IdP.

Vous devez également créer des règles de réclamation appropriées dans votre IdP, qui spécifient en AWS tant que partie de confiance. Lorsque l'IdP envoie une réponse SAML au point de AWS terminaison, elle inclut une assertion SAML contenant une ou plusieurs revendications. Une demande contient des informations sur l'utilisateur et ses groupes. Une règle de demande mappe ces informations à des attributs SAML. Cela vous permet de vous assurer que les réponses d'authentification SAML de votre IdP contiennent les attributs nécessaires utilisés dans AWS les politiques IAM pour vérifier les autorisations des utilisateurs fédérés. Pour plus d'informations, consultez les rubriques suivantes :

- Vue d'ensemble du rôle permettant d'autoriser l'accès fédéré par SAML à vos ressources AWS. Cette rubrique décrit les clés spécifiques à SAML dans les politiques IAM et explique comment les utiliser pour limiter les autorisations des utilisateurs fédérés SAML.
- Configurer les assertions SAML pour la réponse d'authentification. Cette rubrique explique comment configurer des demandes SAML contenant des informations sur l'utilisateur. Les demandes sont regroupées dans une assertion SAML et intégrées dans la réponse SAML qui est envoyée à AWS. Vous devez vous assurer que les informations requises par AWS les politiques sont incluses dans l'assertion SAML sous une forme AWS reconnaissable et utilisable.
- Intégrez des fournisseurs de solutions SAML tiers avec AWS. Cette rubrique fournit des liens vers la documentation fournie par des organisations tierces sur la manière d'intégrer des solutions d'identité à AWS.

> **Note**
>
> Pour améliorer la résilience de la fédération, nous vous recommandons de configurer votre IdP et votre fédération AWS pour prendre en charge plusieurs points de terminaison de connexion SAML. Pour plus de détails, consultez l'article du blog sur la AWS sécurité Comment utiliser les points de terminaison SAML régionaux pour le basculement.

Intégrez des fournisseurs de solutions SAML tiers avec AWS

> **Note**
>
> Nous vous recommandons de demander à vos utilisateurs humains d'utiliser des informations d'identification temporaires lors de l'accès AWS. Avez-vous envisagé d'en utiliser AWS IAM Identity Center ? Vous pouvez utiliser IAM Identity Center pour gérer de manière centralisée l'accès à plusieurs comptes Comptes AWS et fournir aux utilisateurs un accès par authentification unique protégé par le MFA à tous les comptes qui leur sont attribués à partir d'un seul endroit. Avec IAM Identity Center, vous pouvez créer et gérer les identités des utilisateurs dans IAM Identity Center ou vous connecter facilement à votre fournisseur d'identité compatible SAML 2.0 existant. Pour plus d'informations, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Les liens suivants vous aident à configurer des solutions de fournisseur d'identité (IdP) SAML 2.0 tiers pour qu'elles fonctionnent avec AWS la fédération.

> **Tip**
>
> AWS Les ingénieurs de support peuvent aider les clients disposant de plans de support professionnels et professionnels à effectuer certaines tâches d'intégration impliquant des logiciels tiers. Pour obtenir la liste actuelle des plateformes et applications prises en charge, veuillez consulter la section What third-party software is supported? (Quels sont les logiciels tiers pris en charge ?) dans les FAQ sur AWS Support.

Solution	En savoir plus
Auth0	Intégration à Amazon Web Services — Cette page du site Web de documentation Auth0 contient des liens vers des ressources qui décrivent comment configurer l'authent ification unique (SSO) avec le AWS Management Console et inclut un exemple. JavaScript Vous pouvez configurer Auth0 pour transmettre les balises de session. Pour plus d'informations, voir Auth0 annonce un partenariat avec AWS for IAM Session Tags.

Solution	En savoir plus
Microsoft Entra	Tutoriel : intégration de Microsoft Entra SSO à l'accès AWS à compte unique — Ce didacticiel disponible sur le site Web de Microsoft explique comment configurer Microsoft Entra (anciennement Azure AD) en tant que fournisseur d'identité (IdP) à l'aide de la fédération SAML.
Centrify	Configurer Centrify et utiliser SAML pour le SSO pour AWS — Cette page du site Web de Centrify explique comment configurer Centrify pour utiliser le SAML pour le SSO pour. AWS
CyberArk	Configurez CyberArkpour fournir un accès Amazon Web Services (AWS) aux utilisateurs qui se connectent via l'authentification unique (SSO) SAML depuis le CyberArk portail utilisateur.
ForgeRock	La plateforme ForgeRock d'identité s'intègre à AWS. Vous pouvez configurer ForgeRock pour transmettre les balises de session. Pour de plus amples informations, veuillez consulter Attribute Based Access Control for Amazon Web Services.
Google Workspace	Application cloud Amazon Web Services — Cet article du site d'aide aux administrateurs de Google Workspace explique comment configurer Google Workspace en tant qu'IdP SAML 2.0 AWS avec comme fournisseur de services.
IBM	Vous pouvez configurer IBM pour transmettre des balises de session. Pour plus d'informations, consultez IBM Cloud Identity IDaaS, l'un des premiers à prendre en charge les balises de AWS session.

Solution	En savoir plus
JumpCloud	Octroi d'accès via des rôles IAM pour l'authentification unique (SSO) avec Amazon AWS — Cet article du site JumpCloud Web explique comment configurer et activer l'authentification unique en fonction des rôles IAM pour. AWS
Matrix42	MyWorkspace Guide de démarrage — Ce guide explique comment intégrer les services d' AWS identité à MyWorkspace Matrix42.
Microsoft Active Directory Federation Services (AD FS)	Remarques sur le terrain : intégration du service de fédération Active Directory à AWS IAM Identity Center — Ce billet du blog sur AWS l'architecture explique le flux d'authentification entre AD FS et AWS IAM Identity Center (IAM Identity Center). IAM Identity Center prend en charge la fédération d'identité avec SAML 2.0, ce qui permet l'intégration avec les solutions AD FS. Les utilisateurs peuvent se connecter au portail IAM Identity Center avec leurs informations d'identification d'entreprise, ce qui réduit les coûts administratifs liés à la gestion des informations d'identification distinctes sur IAM Identity Center. Vous pouvez également configurer AD FS pour transmettre des balises de session. Pour de plus amples informations, veuillez consulter Use attribute-based access control with AD FS to simplify IAM permissions management.
miniOrange	SSO pour AWS — Cette page du site Web de MiniOrange explique comment établir un accès sécurisé AWS pour les entreprises et un contrôle total de l'accès aux AWS applications.

Solution	En savoir plus
Okta	Integrating the Amazon Web Services Command Line Interface Using Okta : sur cette page du site d'assistance Okta, vous apprenez à configurer Okta en vue d'une utilisation avec AWS. Vous pouvez configurer Okta pour transmettre des balises de session. Pour plus d'informations, consultez Okta et AWS son partenariat pour simplifier l'accès via des balises de session.
Okta	AWS Fédération de comptes : cette section du site Web d'Okta explique comment configurer et activer IAM Identity Center pour. AWS
OneLogin	Dans la OneLoginbase de connaissances, **`SAML AWS`** recherchez une liste d'articles expliquant comment configurer les fonctionnalités d'IAM Identity Center entre OneLogin et AWS pour des scénarios à rôle unique et à rôles multiples. Vous pouvez configurer OneLogin pour transmettre les balises de session. Pour plus d'informations, voir OneLogin et Tags de session : contrôle d'accès basé sur les attributs pour les AWS ressources.
Ping Identity	PingFederate AWS Connecteur : consultez les détails du PingFederate AWS connecteur, un modèle de connexion rapide permettant de configurer facilement une connexion d'authentification unique (SSO) et de provisionnement. Lisez la documentation et téléchargez le dernier PingFederate AWS connecteur pour les intégrations avec AWS. Vous pouvez configurer Ping Identity pour transmettre des balises de session. Pour de plus amples informations, veuillez consulter Announcing Ping Identity Support for Attribute-Based Access Control in AWS.

Solution	En savoir plus
RadiantLogic	Partenaires technologiques de Radiant Logic — Le service d'identité RadiantOne fédéré de Radiant Logic s'intègre AWS pour fournir un hub d'identité pour le SSO basé sur SAML.
RSA	Le guide de mise en œuvre d'Amazon Web Services - RSA Ready fournit des conseils pour l'intégration AWS et le RSA. Pour plus d'informations sur la configuration SAML, consultez le guide d'implémentation Amazon Web Services - Configuration SSO SAML My Page - RSA Ready.
Salesforce.com	Comment configurer l'authentification unique depuis Salesforce vers AWS : cet article pratique sur le site destiné aux développeurs de Salesforce.com explique comment configurer un fournisseur d'identité (IdP) dans Salesforce et le configurer en tant que fournisseur de services. AWS
SecureAuth	AWS - SSO SecureAuth SAML — Cet article du site SecureAuth Web explique comment configurer l'intégration SAML AWS pour une appliance. SecureAuth
Shibboleth	Comment utiliser Shibboleth pour l'authentification unique AWS Management Console— Cet article du blog sur la AWS sécurité fournit un step-by-step didacticiel sur la façon de configurer Shibboleth et de le configurer en tant que fournisseur d'identité pour. AWS Vous pouvez configurer Shibboleth pour transmettre des balises de session.

Pour plus de détails, consultez la page des partenaires IAM sur le AWS site Web.

Configurer les assertions SAML pour la réponse d'authentification

Après avoir vérifié l'identité d'un utilisateur dans votre organisation, le fournisseur d'identité externe (IdP) envoie une réponse d'authentification au point de terminaison AWS SAML à l'adresse. `https://region-code.signin.aws.amazon.com/saml` Pour obtenir la liste des

remplacements potentiels de *`region-code`*, consultez la colonne Region (Région) des AWS Sign-In endpoints (Points de terminaison de connexion). La réponse est une demande POST incluant un jeton SAML conforme à la norme de liaison HTTP POST pour SAML 2.0 et contenant les éléments suivants, ou demandes. Vous les configurez dans votre IdP compatible avec SAML. Pour obtenir des instructions sur la procédure visant à entrer ces demandes, consultez la documentation de votre IdP.

Lorsque l'IdP envoie la réponse contenant les demandes à AWS, de nombreuses demandes entrantes sont associées à des clés de AWS contexte. Ces clés de contexte peuvent être vérifiées dans les politiques IAM à l'aide de l'élément `Condition`. Pour obtenir une liste des mappages disponibles, consultez la section Mappage des attributs SAML avec des clés AWS contextuelles de politique de confiance.

`Subject` et **`NameID`**

L'extrait suivant en présente un exemple. Substituez vos propres valeurs avec les valeurs marquées. Il doit y avoir exactement 1 élément `SubjectConfirmation` avec un élément `SubjectConfirmationData` comprenant à la fois l'attribut `NotOnOrAfter` et un attribut `Recipient`. Ces attributs incluent une valeur qui doit correspondre au AWS point de terminaison`https://`*`region-code`*`.signin.aws.amazon.com/saml`. Pour obtenir la liste des valeurs possibles de *`region-code`*, consultez la colonne Region (Région) des AWS Sign-In endpoints (Points de terminaison de connexion). Pour la AWS valeur, vous pouvez également utiliser`https://signin.aws.amazon.com/static/saml`, comme indiqué dans l'exemple suivant.

Les éléments `NameID` peuvent avoir la valeur persistante, transitoire ou consister en l'URI de format complet tel que fourni par la solution d'IdP. La valeur persistante indique que la valeur dans `NameID` reste la même pour l'utilisateur entre les sessions. Si la valeur est transitoire, l'utilisateur a une valeur `NameID` différente pour chaque session. Les interactions avec authentification unique prennent en charge les types d'identifiants suivants :

- `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:transient`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos`

- `urn:oasis:names:tc:SAML:2.0:nameid-format:entity`

```
<Subject>
  <NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">_cbb88bf52c2510eabe00c1642d4643f41430fe25e3</NameID>
  <SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
    <SubjectConfirmationData NotOnOrAfter="2013-11-05T02:06:42.876Z"
 Recipient="https://signin.aws.amazon.com/saml"/>
  </SubjectConfirmation>
</Subject>
```

> **Important**
>
> La clé de contexte `saml:aud` vient de l'attribut destinataire SAML, qui est l'équivalent SAML du champ d'audience d'OIDC, par exemple `accounts.google.com:aud`.

Attribut SAML **`PrincipalTag`**

(Facultatif) Vous pouvez utiliser un élément `Attribute` dont l'attribut `Name` est défini sur `https://aws.amazon.com/SAML/Attributes/PrincipalTag:{TagKey}`. Cet élément vous permet de transmettre des attributs en tant que balises de session dans l'assertion SAML. Pour de plus amples informations sur les balises de session, veuillez consulter Transmission des balises de session AWS STS.

Pour transmettre des attributs en tant que balises de session, incluez l'élément `AttributeValue` qui spécifie la valeur de la balise. Par exemple, pour transmettre les paires clé-valeur de balise `Project` = `Marketing` et `CostCenter` = `12345`, utilisez l'attribut suivant. Incluez un élément `Attribute` distinct pour chaque balise.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Marketing</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
```

Pour définir les balises ci-dessus comme transitives, incluez un autre élément `Attribute` dont l'attribut `Name` est défini sur `https://aws.amazon.com/SAML/Attributes/`

`TransitiveTagKeys`. Il s'agit d'un attribut facultatif à valeurs multiples qui définit vos balises de session comme transitives. Les balises transitives persistent lorsque vous utilisez la session SAML pour endosser un autre rôle dans AWS. Ceci est connu sous le nom de chaînage de rôles. Par exemple, pour définir les balises `Principal` et `CostCenter` comme transitives, utilisez l'attribut suivant pour spécifier les clés.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>CostCenter</AttributeValue>
</Attribute>
```

Attribut SAML **`Role`**

Vous pouvez utiliser un élément `Attribute` avec l'attribut `Name` défini sur `https://aws.amazon.com/SAML/Attributes/Role` Cet élément contient un ou plusieurs éléments `AttributeValue` qui répertorient le fournisseur d'identité et le rôle IAM auxquels l'utilisateur est mappé par votre IdP. Le rôle IAM et le fournisseur d'identité IAM sont spécifiés sous la forme d'une paire d'ARN séparés par des virgules au même format que les `PrincipalArn` paramètres `RoleArn` et transmis au SAML. AssumeRoleWith Cet élément doit contenir au moins une paire de fournisseur de rôles (élément `AttributeValue`) et peut contenir plusieurs paires. Si l'élément contient plusieurs paires, l'utilisateur doit choisir le rôle à endosser lorsqu'il utilise WebSSO afin de se connecter à la AWS Management Console.

> **Important**
>
> La valeur de l'attribut `Name` dans la balise `Attribute` est sensible à la casse. Il doit être défini sur `https://aws.amazon.com/SAML/Attributes/Role` précisément.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/Role">
  <AttributeValue>arn:aws:iam::account-number:role/role-name1,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name2,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name3,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
</Attribute>
```

Attribut SAML **RoleSessionName**

Vous pouvez utiliser un élément `Attribute` avec l'attribut `Name` défini sur `https://aws.amazon.com/SAML/Attributes/RoleSessionName` Cet élément contient un élément `AttributeValue` qui fournit un identifiant pour les informations d'identification temporaires qui sont émises lorsque le rôle est endossé. Vous pouvez l'utiliser pour associer les informations d'identification temporaires à l'utilisateur qui utilise votre application. Cet élément est utilisé pour afficher les informations utilisateur dans le AWS Management Console. La valeur de l'élément `AttributeValue` doit comporter entre 2 et 64 caractères. Elle ne peut contenir que des caractères alphanumériques, des traits de soulignement et les caractères suivants : . , + = @ - (tiret). Il ne doit pas contenir d'espace. La valeur est généralement un ID utilisateur (`johndoe`) ou une adresse e-mail (`johndoe@example.com`). Il ne peut pas contenir d'espace, comme dans le nom complet de l'utilisateur (`John Doe`).

> **Important**
>
> La valeur de l'attribut `Name` dans la balise `Attribute` est sensible à la casse. Il doit être défini sur `https://aws.amazon.com/SAML/Attributes/RoleSessionName` précisément.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/RoleSessionName">
  <AttributeValue>user-id-name</AttributeValue>
</Attribute>
```

Attribut SAML **SessionDuration**

(Facultatif) Vous pouvez utiliser un élément `Attribute` dont l'attribut `Name` est défini sur `https://aws.amazon.com/SAML/Attributes/SessionDuration"`. Cet élément contient un `AttributeValue` élément qui indique pendant combien de temps l'utilisateur peut accéder au AWS Management Console avant de devoir demander de nouvelles informations d'identification temporaires. La valeur est un nombre entier représentant le nombre de secondes pour la session. La valeur peut être comprise entre 900 secondes (15 minutes) et 43 200 secondes (12 heures). Si cet attribut n'est pas présent, la durée des informations d'identification est d'une d'heure (valeur par défaut du paramètre `DurationSeconds` de l'API `AssumeRoleWithSAML`).

Pour utiliser cet attribut, vous devez configurer le fournisseur SAML de manière à fournir un accès AWS Management Console par authentification unique au point de terminaison Web de connexion à

la console à l'adresse. `https://region-code.signin.aws.amazon.com/saml` Pour obtenir la liste des valeurs possibles de *region-code*, consultez la colonne Region (Région) des AWS Sign-In endpoints (Points de terminaison de connexion). Vous pouvez éventuellement utiliser l'URL suivante : `https://signin.aws.amazon.com/static/saml`. Notez que cet attribut prolonge les sessions uniquement à la AWS Management Console. Il ne peut pas prolonger la durée de vie des autres informations d'identification. Cependant, s'il est présent dans un appel d'API `AssumeRoleWithSAML`, il peut être utilisé pour raccourcir la durée de la session. La durée de vie par défaut des informations d'identification renvoyées par l'appel est de 60 minutes.

Notez également que si un attribut `SessionNotOnOrAfter` est également défini, la valeur la plus faible des deux attributs `SessionDuration` ou `SessionNotOnOrAfter` établit la durée maximale de la session de console.

Lorsque vous activez les sessions de console avec une durée prolongée, le risque de divulgation des informations d'identification augmente. Pour vous aider à réduire ce risque, vous pouvez immédiatement désactiver les sessions de console actives pour un rôle en sélectionnant Revoke Sessions (Révoquer les sessions) sur la page Role Summary (Résumé du rôle) de la console IAM. Pour plus d'informations, veuillez consulter Révocation des informations d'identification de sécurité temporaires d'un rôle IAM.

> **Important**
>
> La valeur de l'attribut `Name` dans la balise `Attribute` est sensible à la casse. Il doit être défini sur `https://aws.amazon.com/SAML/Attributes/SessionDuration` précisément.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SessionDuration">
  <AttributeValue>1800</AttributeValue>
</Attribute>
```

Attribut SAML **`SourceIdentity`**

(Facultatif) Vous pouvez utiliser un élément `Attribute` dont l'attribut `Name` est défini sur `https://aws.amazon.com/SAML/Attributes/SourceIdentity`. Cet élément contient un élément `AttributeValue` qui fournit un identifiant pour la personne ou l'application qui utilise un rôle IAM. La valeur de l'identité source est conservée lorsque vous utilisez la session SAML pour assumer un autre rôle AWS connu sous le nom de chaînage de rôles. La valeur de l'identité source est présente

dans la demande pour chaque action effectuée durant la session de rôle. La valeur définie ne peut pas être modifiée durant la session de rôle. Les administrateurs peuvent ensuite utiliser AWS CloudTrail les journaux pour surveiller et auditer les informations d'identité source afin de déterminer qui a effectué des actions avec des rôles partagés.

La valeur de l'élément `AttributeValue` doit comporter entre 2 et 64 caractères. Elle ne peut contenir que des caractères alphanumériques, des traits de soulignement et les caractères suivants : . , + = @ - (tiret). Il ne doit pas contenir d'espace. La valeur est généralement un attribut associé à l'utilisateur, comme un id utilisateur (`johndoe`) ou une adresse e-mail (`johndoe@example.com`). Il ne peut pas contenir d'espace, comme dans le nom complet de l'utilisateur (`John Doe`). Pour de plus amples informations sur l'utilisation de l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

> **Important**
>
> Si votre assertion SAML est configurée pour utiliser l'attribut `SourceIdentity`, votre politique d'approbation de rôle doit également inclure l'action `sts:SetSourceIdentity`, sinon l'opération endosser le rôle échouera. Pour de plus amples informations sur l'utilisation de l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

Pour transmettre un attribut d'identité source, incluez l'élément `AttributeValue` qui spécifie la valeur de l'identité source. Par exemple, pour transmettre l'identité source `DiegoRamirez`, utilisez l'attribut suivant.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
  <AttributeValue>DiegoRamirez</AttributeValue>
```

Mappage des attributs SAML avec des clés AWS contextuelles de politique de confiance

Les tableaux de cette section répertorient les attributs SAML utilisés couramment et leur correspondance avec les clés de contexte de condition de politique d'approbation dans AWS. Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés aux valeurs incluses dans les assertions qui accompagnent une demande d'accès SAML.

> **Important**
>
> Ces clés sont disponibles uniquement dans les politiques d'approbation IAM (politiques qui déterminent qui peut endosser un rôle) et ne sont pas applicables aux politiques d'autorisations.

Dans le tableau des attributs eduPerson et eduOrg, les valeurs sont saisies sous forme de chaînes ou de listes de chaînes. Pour les valeurs de chaînes, vous pouvez tester ces valeurs dans des politiques d'approbation IAM à l'aide des conditions `StringEquals` ou `StringLike`. Concernant les valeurs qui contiennent une liste de chaînes, vous pouvez utiliser les opérateurs d'ensemble de politique `ForAnyValue` et `ForAllValues` pour tester les valeurs dans les politiques de confiance.

> **Note**
>
> Vous ne devez inclure qu'une seule réclamation par clé de AWS contexte. Si vous incluez plusieurs demandes, une seule d'entre elles sera mappée.

Attributs eduPerson et eduOrg

Attribut eduPerson ou eduOrg (**`Name`** clé)	Correspond à cette clé AWS contextuelle (**`FriendlyName`** clé)	Type
`urn:oid:1.3.6.1.4.1.5923.1.1.1.1`	`eduPerson Affiliation`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.1.1.2`	`eduPersonNickname`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.1.1.3`	`eduPersonOrgDN`	Chaîne
`urn:oid:1.3.6.1.4.1.5923.1.1.1.4`	`eduPerson OrgUnitDN`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.1.1.5`	`eduPerson PrimaryAf filiation`	Chaîne

Attribut eduPerson ou eduOrg (**`Name`** clé)	Correspond à cette clé AWS contextuelle (**`FriendlyName`** clé)	Type
`urn:oid:1.3.6.1.4.1.5923.1.1.1.6`	`eduPerson PrincipalName`	Chaîne
`urn:oid:1.3.6.1.4.1.5923.1.1.1.7`	`eduPerson Entitlement`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.1.1.8`	`eduPerson PrimaryOrgUnitDN`	Chaîne
`urn:oid:1.3.6.1.4.1.5923.1.1.1.9`	`eduPerson ScopedAff iliation`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1. 1.1.10`	`eduPerson TargetedID`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1. 1.1.11`	`eduPerson Assurance`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.2.1.2`	`eduOrgHomePageURI`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.2.1.3`	`eduOrgIde ntityAuth NPolicyURI`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.2.1.4`	`eduOrgLegalName`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.2.1.5`	`eduOrgSuperiorURI`	Liste de chaînes
`urn:oid:1.3.6.1.4.1.5923.1.2.1.6`	`eduOrgWhi tePagesURI`	Liste de chaînes
`urn:oid:2.5.4.3`	`cn`	Liste de chaînes

Attributs Active Directory

Attribut AD	Cartes correspondant à cette clé AWS contextuelle	Type
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name`	`name`	Chaîne
`http://schemas.xmlsoap.org/claims/CommonName`	`commonName`	Chaîne
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname`	`givenName`	Chaîne
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname`	`surname`	Chaîne
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`	`mail`	Chaîne
`http://schemas.microsoft.com/ws/2008/06/identity/claims/primarygroupsid`	`uid`	Chaîne

Attributs X.500

Attribut X.500	Cartes correspondant à cette clé AWS contextuelle	Type
`2.5.4.3`	`commonName`	Chaîne
`2.5.4.4`	`surname`	Chaîne
`2.4.5.42`	`givenName`	Chaîne
`2.5.4.45`	`x500UniqueIdentifier`	Chaîne
`0.9.2342.19200300100.1.1`	`uid`	Chaîne

Attribut X.500	Cartes correspondant à cette clé AWS contextuelle	Type
`0.9.2342.19200300100.1.3`	`mail`	Chaîne
`0.9.2342.19200300.100.1.45`	`organizationStatus`	Chaîne

Permettre aux utilisateurs fédérés SAML 2.0 d'accéder au AWS Management Console

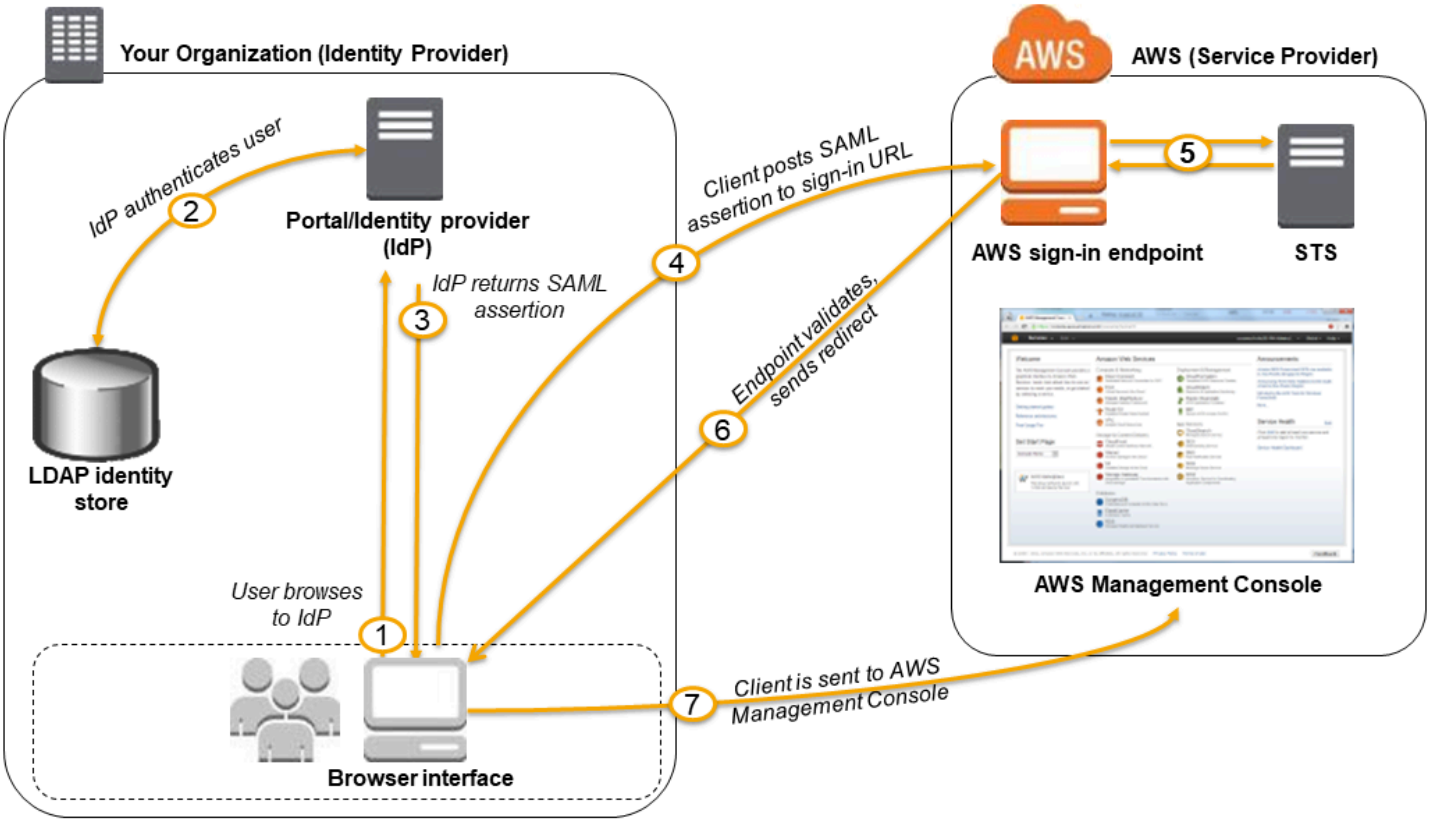
Vous pouvez utiliser un rôle pour configurer votre fournisseur d'identité (IdP) conforme à SAML 2.0 et AWS pour autoriser vos utilisateurs fédérés à accéder au. AWS Management Console Le rôle accorde aux utilisateurs les autorisations nécessaires pour effectuer des tâches dans la console. Pour donner aux utilisateurs fédérés SAML d'autres moyens d'accéder à AWS, veuillez consulter l'une des rubriques suivantes :

- AWS CLI: Assumer un rôle IAM (AWS CLI)
- Outils pour Windows PowerShell : Passer à un rôle IAM (Outils pour Windows PowerShell)
- AWS API : Passage à un rôle IAM (AWS API)

Présentation

Le diagramme suivant illustre le flux d'une authentification unique SAML.

> **Note**
>
> Cette utilisation spécifique de SAML diffère de l'utilisation plus générale illustrée ici Fédération SAML 2.0 car ce flux de travail ouvre le fichier AWS Management Console au nom de l'utilisateur. Cette procédure requiert l'utilisation du point de terminaison SSO AWS au lieu d'un appel direct de l'API `AssumeRoleWithSAML`. Le point de terminaison appelle l'API pour l'utilisateur, puis retourne une URL qui redirige automatiquement le navigateur de l'utilisateur vers AWS Management Console.



Le diagramme suivant illustre les étapes suivantes :

1. L'utilisateur se rend sur le portail de l'organisation, puis choisit d'accéder à l'interface AWS Management Console. Dans votre organisation, le portail est généralement une fonction de votre IdP qui gère l'échange de confiance entre votre organisation et. AWS Par exemple, dans les services Active Directory Federation Services, l'URL du portail est : `https://ADFSServiceName/adfs/ls/IdpInitiatedSignOn.aspx`
2. Le portail vérifie l'identité de l'utilisateur dans l'organisation.
3. Il génère ensuite une réponse d'authentification SAML contenant des assertions qui identifient l'utilisateur et incluent des attributs spécifiques à celui-ci. Vous pouvez également configurer votre IdP pour inclure un attribut d'assertion SAML appelé `SessionDuration` qui spécifie la durée de validité de la session de console. Vous pouvez également configurer le fournisseur d'identité pour qu'il transmette des attributs en tant que balises de session. Le portail envoie cette réponse au navigateur client.
4. Le navigateur client est redirigé vers le point de terminaison d'authentification AWS unique et publie l'assertion SAML.
5. Le point de terminaison demande des informations d'identification de sécurité temporaires pour le compte de l'utilisateur et crée une URL de connexion à la console qui utilise ces informations.

6. AWS renvoie l'URL de connexion au client sous forme de redirection.
7. Le navigateur client est redirigé vers AWS Management Console. Si la réponse d'authentification SAML inclut des attributs associés à plusieurs rôles IAM, l'utilisateur est d'abord invité à sélectionner le rôle utilisé pour accéder à la console.

Du point de vue de l'utilisateur, le processus se déroule de manière transparente : l'utilisateur commence sur le portail interne de votre organisation et se retrouve sur le portail AWS Management Console, sans jamais avoir à fournir d'informations d' AWS identification.

Consultez les sections suivantes pour une présentation de la configuration de ce comportement, ainsi que des liens vers des procédures détaillées.

Configurez votre réseau en tant que fournisseur SAML pour AWS

Au sein du réseau de votre organisation, vous configurez votre base d'identités (par ex., Windows Active Directory) afin qu'il fonctionne de concert avec un fournisseur d'identité (IdP) SAML tel que Windows Active Directory Federation Services, Shibboleth, etc. À l'aide de l'IdP, vous générez un document de métadonnées qui décrit votre organisation en tant que fournisseur d'identité et inclut des clés d'authentification. Vous configurez également le portail de votre organisation pour acheminer les demandes des utilisateurs vers le point de terminaison AWS SAML AWS Management Console à des fins d'authentification à l'aide d'assertions SAML. La configuration de l'IdP pour la génération d'un fichier metadata.xml varie en fonction de l'IdP. Pour savoir comment procéder, reportez-vous à la documentation de votre IdP, ou consultez Intégrez des fournisseurs de solutions SAML tiers avec AWS pour obtenir des liens vers la documentation web de nombreux fournisseurs SAML pris en charge.

Créer un fournisseur SAML dans IAM

Ensuite, vous vous connectez à la console IAM AWS Management Console et vous y accédez. Dans la console, vous créez un nouveau fournisseur SAML, c'est-à-dire une entité IAM qui contient des informations sur le fournisseur d'identité de votre organisation. Au cours du processus, vous téléchargez le document de métadonnées généré par le logiciel de l'IdP de votre organisation à l'étape précédente. Pour plus de détails, consultez Création d'un fournisseur d'identité SAML dans IAM.

Configurez les autorisations AWS pour vos utilisateurs fédérés

L'étape suivante consiste à créer un rôle IAM qui établit une relation d'approbation entre IAM et le fournisseur d'identités de votre organisation. Ce rôle doit identifier votre fournisseur d'identité en tant

que principal (entité de confiance) pour les besoins de la fédération. Le rôle définit également ce que les utilisateurs authentifiés par l'IdP de votre organisation sont autorisés à faire. AWS Vous pouvez créer ce rôle à l'aide de la console IAM. Lorsque vous créez la politique d'approbation qui indique qui peut endosser le rôle, vous indiquez le fournisseur SAML que vous avez créé précédemment dans IAM. Vous indiquez également un ou plusieurs attributs SAML auxquels un utilisateur doit correspondre pour être autorisé à endosser le rôle. Par exemple, vous pouvez spécifier que seuls les utilisateurs dont la valeur `eduPersonOrgDN` SAML est `ExampleOrg` sont autorisés à se connecter. L'assistant de rôle ajoute automatiquement une condition pour tester l'attribut `saml:aud` afin de vérifier que le rôle est endossé uniquement pour la connexion à AWS Management Console. La politique d'approbation du rôle peut se présenter comme suit :

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/
ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {"StringEquals": {
      "saml:edupersonorgdn": "ExampleOrg",
      "saml:aud": "https://signin.aws.amazon.com/saml"
    }}
  }]
}
```

> **Note**
>
> Les IdP SAML utilisés dans une politique d'approbation de rôle doivent se trouver dans le même compte que le rôle.

Vous pouvez inclure des points de terminaison régionaux pour l'attribut `saml:aud` à l'adresse `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`. Pour obtenir la liste des valeurs possibles de *`region-code`*, consultez la colonne Region (Région) des AWS Sign-In endpoints (Points de terminaison de connexion).

Dans la politique d'autorisation du rôle, vous spécifiez les autorisations comme pour tout autre rôle, utilisateur ou groupe. Par exemple, si les utilisateurs de votre organisation sont autorisés à administrer les instances Amazon EC2, vous autorisez explicitement les actions Amazon EC2 dans la

politique d'autorisation. Pour ce faire, vous pouvez affecter une politique gérée comme Amazon EC2 Full Access.

Pour plus d'informations sur la création d'un rôle pour un IdP SAML, consultez Création d'un rôle pour la fédération SAML 2.0 (console).

Terminer la configuration et créer des assertions SAML

Indiquez à votre IdP SAML AWS qu'il s'agit de votre fournisseur de services en installant `saml-metadata.xml` le fichier trouvé `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml` sur ou. `https://signin.aws.amazon.com/static/saml-metadata.xml` Pour obtenir la liste des valeurs possibles de *`region-code`*, consultez la colonne Region (Région) des AWS Sign-In endpoints (Points de terminaison de connexion).

L'installation du fichier est différente selon votre IdP. Certains fournisseurs vous permettent d'entrer l'URL, après quoi ils récupèrent et installent le fichier pour vous. D'autres fournisseurs exigent que vous téléchargiez le fichier à partir de l'URL afin de le fournir en tant que fichier local. Pour plus d'informations, reportez-vous à la documentation de votre IdP, ou consultez Intégrez des fournisseurs de solutions SAML tiers avec AWS pour obtenir des liens vers la documentation web de nombreux fournisseurs SAML pris en charge.

Vous devez également configurer les informations que l'IdP doit transmettre à AWS en tant qu'attributs SAML dans le cadre de la réponse d'authentification. La plupart de ces informations apparaissent AWS sous forme de clés contextuelles de condition que vous pouvez évaluer dans vos politiques. Ces clés de condition garantissent que seuls les utilisateurs autorisés dans les contextes appropriés sont autorisés à accéder à vos ressources AWS . Vous pouvez spécifier des fenêtres horaires qui limitent l'utilisation de la console. Vous pouvez également spécifier la durée maximale (jusqu'à 12 heures) pendant laquelle les utilisateurs peuvent accéder à la console avant d'avoir à actualiser leurs informations d'identification. Pour plus d'informations, consultez Configurer les assertions SAML pour la réponse d'authentification.

Informations d'identification de sécurité temporaires dans IAM

Vous pouvez utiliser le AWS Security Token Service (AWS STS) pour créer et fournir à des utilisateurs de confiance des informations d'identification de sécurité temporaires qui peuvent contrôler l'accès à vos AWS ressources. Les informations d'identification temporaires fonctionnent de manière presque identique aux informations d'identification des clés d'accès à long terme, à quelques différences près :

- Les informations d'identification de sécurité temporaires sont à court terme, comme leur nom l'indique. Elles peuvent être configurées pour être valides de quelques minutes à plusieurs heures. Une fois les informations d'identification AWS expirées, il ne les reconnaît plus ou n'autorise aucun type d'accès à partir des demandes d'API effectuées avec elles.
- Les informations d'identification de sécurité temporaires ne sont pas stockées avec l'utilisateur, mais sont générées automatiquement et fournies à l'utilisateur sur demande. Lorsque (ou même avant) les informations d'identification de sécurité temporaires arrivent à expiration, l'utilisateur peut en demander de nouvelles, tant qu'il y est autorisé.

Par conséquent, les informations d'identification temporaires présentent les avantages suivants par rapport aux informations d'identification à long terme :

- Il n'est pas nécessaire de distribuer ou d'intégrer des informations de AWS sécurité à long terme dans une application.
- Vous pouvez donner accès à vos AWS ressources aux utilisateurs sans avoir à définir leur AWS identité. Les informations d'identification temporaires sont à la base de la fédération des rôles et des identités.
- Les informations d'identification de sécurité temporaires ont une durée de vie limitée. Il n'est donc pas nécessaire de les mettre à jour ou de les révoquer explicitement lorsqu'elles deviennent obsolètes. Une fois que les informations d'identification de sécurité temporaires arrivent à expiration, elles ne peuvent pas être réutilisées. Vous pouvez spécifier le délai de validité des informations d'identification, jusqu'à une certaine limite.

AWS STS et AWS régions

Les informations d'identification de sécurité temporaires sont générées par AWS STS. Par défaut, AWS STS il s'agit d'un service global avec un seul point de terminaison à`https://sts.amazonaws.com`. Toutefois, vous pouvez également choisir d'effectuer des appels d' AWS STS API vers des points de terminaison situés dans toute autre région prise en charge. Cela permet de réduire la latence (décalage serveur) en envoyant des demandes aux serveurs situés dans une région géographiquement plus proche de vous. Quelle que soit la région d'où proviennent vos informations d'identification, elles fonctionnent dans le monde entier. Pour plus d'informations, veuillez consulter Gérer AWS STS dans un Région AWS.

Scénarios courants d'informations d'identification temporaires

Les informations d'identification temporaires sont utiles dans des scénarios impliquant la fédération d'identité, la délégation, l'accès entre comptes et les rôles IAM.

Fédération des identités

Vous pouvez gérer les identités de vos utilisateurs dans un système externe AWS et accorder aux utilisateurs qui se connectent à partir de ces systèmes l'accès pour effectuer AWS des tâches et accéder à vos AWS ressources. IAM prend en charge deux types de fédération d'identité. Dans les deux cas, les identités sont stockées à l'extérieur de AWS. La distinction réside dans l'endroit où se trouve le système externe : dans votre centre de données ou dans un tiers externe sur le Web. Pour plus d'informations sur les fournisseurs d'identité externes, consultez Fournisseurs d'identité et fédération.

- Fédération SAML : vous pouvez authentifier les utilisateurs sur le réseau de votre organisation, puis leur fournir un accès AWS sans leur créer de nouvelles AWS identités ni leur demander de se connecter avec des informations d'identification différentes. C'est ce que l'on appelle l'approche d'authentification unique pour l'accès temporaire. AWS STS prend en charge les normes ouvertes telles que le langage SAML (Security Assertion Markup Language) 2.0, avec lequel vous pouvez utiliser Microsoft AD FS pour tirer parti de votre Microsoft Active Directory. SAML 2.0 vous permet également de gérer votre propre solution pour fédérer les identités des utilisateurs. Pour plus d'informations, consultez Fédération SAML 2.0.
  - Courtier de fédération personnalisé : vous pouvez utiliser le système d'authentification de votre organisation pour accorder l'accès aux AWS ressources. Pour obtenir un exemple de scénario, consultez Activation de l'accès à la AWS console par un courtier d'identité personnalisé.
  - Federation using SAML 2.0 (Fédération à l'aide de SAML 2.0) : vous pouvez utiliser le système d'authentification de votre organisation et SAML pour accorder l'accès aux ressources AWS . Pour obtenir des informations et un exemple de scénario, consultez Fédération SAML 2.0.
- Fédération OpenID Connect (OIDC) — Vous pouvez autoriser les utilisateurs à se connecter en utilisant un fournisseur d'identité tiers connu tel que Login with Amazon, Facebook, Google ou tout autre fournisseur compatible OIDC 2.0 pour votre application mobile ou Web. Vous n'avez pas besoin de créer de code de connexion personnalisé ni de gérer vos propres identités d'utilisateur. L'utilisation de la fédération OIDC vous aide à Compte AWS garantir votre sécurité, car vous n'avez pas à distribuer des informations de sécurité à long terme, telles que des clés d'accès utilisateur IAM, avec votre application. Pour plus d'informations, consultez Fédération OIDC.

AWS STS La fédération OIDC prend en charge Login with Amazon, Facebook, Google et tout autre fournisseur d'identité compatible avec OpenID Connect (OIDC).

> **Note**
>
> Pour les applications mobiles, nous vous recommandons d'utiliser Amazon Cognito. Vous pouvez utiliser ce service avec AWS les SDK pour le développement mobile afin de créer des identités uniques pour les utilisateurs et de les authentifier afin de sécuriser l'accès à vos AWS ressources. Amazon Cognito prend en charge les mêmes fournisseurs d'identité AWS STS, prend également en charge l'accès non authentifié (invité) et vous permet de migrer les données utilisateur lorsqu'un utilisateur se connecte. Amazon Cognito fournit également des opérations d'API pour la synchronisation des données utilisateur afin de les conserver à mesure que les utilisateurs passent d'un périphérique à l'autre. Pour plus d'informations, consultez Authentification avec Amplify dans la documentation Amplify.

Rôles pour l'accès entre comptes

De nombreuses organisations gèrent plusieurs Compte AWS. À l'aide des rôles et de l'accès entre comptes, vous pouvez définir des identités utilisateur dans un compte et utiliser ces identités pour accéder aux ressources AWS dans d'autres comptes qui appartiennent à votre organisation. Cette procédure s'appelle la délégation pour un accès temporaire. Pour de plus amples informations sur la création de rôles entre comptes, veuillez consulter la rubrique Création d'un rôle pour la délégation d'autorisations à un utilisateur IAM.. Pour savoir si les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, veuillez consulter Qu'est-ce que l'Analyseur d'accès IAM ?.

Rôles pour Amazon EC2

Si vous exécutez des applications sur des instances Amazon EC2 et que ces applications doivent accéder à des ressources AWS , vous pouvez fournir des informations d'identification de sécurité temporaires à vos instances lorsque vous les lancez. Ces informations d'identification de sécurité temporaires sont disponibles pour toutes les applications qui s'exécutent sur l'instance, vous n'avez donc pas besoin de stocker des informations d'identification à long terme sur l'instance. Pour plus d’informations, consultez Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2.

Autres AWS services

Vous pouvez utiliser des informations d'identification de sécurité temporaires pour accéder à la plupart AWS des services. Pour obtenir une liste des services qui acceptent les informations d'identification de sécurité temporaires, consultez la section AWS services qui fonctionnent avec IAM.

Demande d'informations d'identification temporaires de sécurité

Pour demander des informations d'identification de sécurité temporaires, vous pouvez utiliser les opérations AWS Security Token Service (AWS STS) dans l' AWS API. Il s'agit notamment d'opérations visant à créer et à fournir à des utilisateurs de confiance des informations d'identification de sécurité temporaires qui peuvent contrôler l'accès à vos AWS ressources. Pour plus d'informations sur AWS STS, voirInformations d'identification de sécurité temporaires dans IAM. Pour en savoir plus sur les différentes méthodes disponibles pour demander des informations d'identification de sécurité temporaires en endossant un rôle, consultez Utilisation de rôles IAM.

Pour appeler les opérations d'API, vous pouvez utiliser l'un des kits SDK AWS. Les kits SDK sont disponibles pour de nombreux langages de programmation et environnements tels que Java, .NET, Python, Ruby, Android et iOS. Ces kits SDK exécutent des tâches telles que la signature cryptographique de vos demandes, les nouvelles tentatives de demande, lorsque cela est nécessaire, ainsi que la gestion des réponses d'erreur. Vous pouvez également utiliser l'API AWS STS Query, qui est décrite dans la référence des AWS Security Token Service API. Enfin, deux outils de ligne de commande prennent en charge les AWS STS commandes : le AWS Command Line Interface, et le AWS Tools for Windows PowerShell.

Les opérations d' AWS STS API créent une nouvelle session avec des informations d'identification de sécurité temporaires qui incluent une paire de clés d'accès et un jeton de session. La paire de clés d'accès comprend un ID de clé d'accès et une clé secrète. Ces informations d'identification permettent aux utilisateurs (ou une application exécutée par l'utilisateur) d'accéder à vos ressources. Vous pouvez créer une session de rôle et transmettre des politiques de session et des balises de session par programmation à l'aide d'opérations d' AWS STS API. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité du rôle et des politiques de session. Pour de amples informations sur les politiques de session, veuillez consulter Politiques de session. Pour de plus amples informations sur les balises de session, veuillez consulter Transmission des balises de session AWS STS.

> **Note**
>
> La taille du jeton de session renvoyé par les opérations d' AWS STS API n'est pas fixe. Nous vous recommandons fortement de ne pas formuler d'hypothèses quant à sa taille maximale. La taille du jeton est en général inférieure à 4 096 octets, mais elle peut varier.

Utilisation AWS STS avec AWS les régions

Vous pouvez envoyer des appels AWS STS d'API à un point de terminaison global ou à l'un des points de terminaison régionaux. En choisissant un point de terminaison plus proche de vous, il est possible de réduire la latence et d'améliorer les performances de vos appels d'API. Vous pouvez également diriger vos appels vers un autre point de terminaison régional si vous n'êtes plus en mesure de communiquer avec le point de terminaison initial. Si vous utilisez l'un des différents AWS SDK, utilisez cette méthode SDK pour spécifier une région avant d'effectuer l'appel d'API. Si vous créez manuellement des demandes d'API HTTP, vous devez les diriger vous-même vers le point de terminaison approprié. Pour plus d'informations, consultez la section AWS STS de Régions et points de terminaison et Gérer AWS STS dans un Région AWS.

Voici les opérations d'API que vous pouvez utiliser pour obtenir des informations d'identification temporaires à utiliser dans votre AWS environnement et vos applications.

AssumeRole : délégation et fédération entre comptes via un broker d'identité personnalisé

Le fonctionnement de l'`AssumeRole`API est utile pour permettre aux utilisateurs IAM existants d'accéder à AWS des ressources auxquelles ils n'ont pas encore accès. Par exemple, l'utilisateur peut avoir besoin d'accéder aux ressources d'un autre Compte AWS. Il est également utile comme moyen d'obtenir temporairement un accès privilégié, par exemple, pour fournir une authentification multi-facteur (MFA). Vous devez appeler cette API à l'aide d'informations d'identification actives. Pour savoir qui peut appeler cette opération, consultez Comparaison des opérations AWS STS d'API. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un utilisateur IAM. et Configuration de l'accès aux API protégé par MFA.

Cet appel doit être effectué à l'aide d'informations AWS de sécurité valides. Lorsque vous effectuez cet appel, vous transmettez les informations suivantes :

- L'Amazon Resource Name (ARN) du rôle que l'application doit endosser.

- (Facultatif) La durée, qui indique la durée des informations d'identification temporaires. Utilisez le paramètre `DurationSeconds` pour spécifier la durée de la session du rôle entre 900 secondes (15 minutes) et la durée de session maximale pour le rôle. Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Affichage du paramètre de durée de session maximale pour un rôle. Si vous ne transmettez pas ce paramètre, les informations d'identification temporaires expirent au bout d'une heure. Le paramètre `DurationSeconds` de cette API est différent du paramètre HTTP `SessionDuration` que vous utilisez pour spécifier la durée de la session d'une console. Utilisez le paramètre HTTP `SessionDuration` de la demande au point de terminaison de fédération d'un jeton de connexion à la console. Pour plus d'informations, veuillez consulter Activation de l'accès à la AWS console par un courtier d'identité personnalisé.
- Nom de la session de rôle. Utilisez cette valeur de chaîne pour identifier la session lorsqu'un rôle est utilisé par différents principaux. Pour des raisons de sécurité, les administrateurs peuvent afficher ce champ dans les journaux AWS CloudTrail pour aider à identifier qui a effectué une action dans AWS. Votre administrateur peut vous demander de spécifier votre nom d'utilisateur IAM comme nom de session lorsque vous endossez le rôle. Pour plus d'informations, consultez `sts:RoleSessionName`.
- (Facultatif) Identité source. Vous pouvez exiger des utilisateurs qu'ils spécifient une identité source lorsqu'ils endossent un rôle. Une fois l'identité source définie, la valeur ne peut pas être modifiée. Elle est présente dans la demande pour toutes les actions qui sont prises durant la session de rôle. La valeur d'identité source persiste entre les sessions de chaînage de rôles. Vous pouvez utiliser les informations d'identité de la source dans AWS CloudTrail les journaux pour déterminer qui a effectué des actions avec un rôle. Pour de plus amples informations sur l'utilisation de l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.
- (Facultatif) Stratégies de session en ligne ou gérées. Ces stratégies limitent les autorisations à partir de la stratégie basée sur l'identité du rôle qui sont attribuées à la session de rôle. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité du rôle et des politiques de session. Les politiques de session ne peuvent pas être utilisées pour accorder plus d'autorisations que celles autorisées par la politique basée sur l'identité du rôle endossé actuellement. Pour de plus amples informations sur les autorisations de session de rôle, veuillez consulter Politiques de session.
- (Facultatif) Balises de session. Vous pouvez endosser un rôle, puis utiliser les informations d'identification temporaires pour effectuer une demande. Lorsque vous le faites, les balises du principal de la session incluent les balises du rôle et les balises de session transmises. Si vous effectuez cet appel à l'aide d'informations d'identification temporaires, la nouvelle session hérite également des balises de session transitive de la session appelante. Pour de plus amples

informations sur les balises de session, veuillez consulter Transmission des balises de session AWS STS.

- (Facultatif) Informations sur MFA. Si la configuration utilise une authentification multi-facteur (MFA), vous incluez l'identificateur d'un dispositif MFA et le code unique fourni par ce dispositif.
- (Facultatif) Une valeur `ExternalId` qui peut être utilisée lorsque vous déléguez l'accès à votre compte à un tiers. Cette valeur permet de s'assurer que seul le tiers spécifié peut accéder au rôle. Pour plus d'informations, veuillez consulter Comment utiliser un identifiant externe lorsque vous accordez l'accès à vos AWS ressources à un tiers.

L'exemple suivant illustre une demande et une réponse qui utilisent `AssumeRole`. Cet exemple de demande endosse le rôle `demo` pour la durée spécifiée avec la politique de session, les balises de session et l'ID externe et l'identité source inclus. La session obtenue est nommée `John-session`.

Example Exemple de demande

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=AssumeRole
&RoleSessionName=John-session
&RoleArn=arn:aws::iam::123456789012:role/demo
&Policy=%7B%22Version%22%3A%222012-10-17%22%2C%22Statement%22%3A%5B%7B%22Sid%22%3A
%20%22Stmt1%22%2C%22Effect%22%3A%20%22Allow%22%2C%22Action%22%3A%20%22s3%3A*%22%2C
%22Resource%22%3A%20%22*%22%7D%5D%7D
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&ExternalId=123ABC
&SourceIdentity=DevUser123
&AUTHPARAMS
```

La valeur de politique indiquée dans l'exemple précédent est la version encodée en URL de la politique suivante :

```
{"Version":"2012-10-17","Statement":
[{"Sid":"Stmt1","Effect":"Allow","Action":"s3:*","Resource":"*"}]}
```

Le paramètre `AUTHPARAMS` de l'exemple est un espace réservé pour votre signature. Une signature est l'information d'authentification que vous devez inclure dans les demandes d'API AWS HTTP. Lors

de la création de demandes d'API, nous vous recommandons d'utiliser les kits SDK AWS, car ceux-ci gèrent diverses tâches pour vous, notamment la signature des demandes. Si vous devez créer et signer des demandes d'API manuellement, consultez la section Signature des AWS demandes à l'aide de la version 4 du Référence générale d'Amazon Web Servicespour savoir comment signer une demande.

Outre les informations d'identification de sécurité temporaires, la réponse inclut l'Amazon Resource Name (ARN) de l'utilisateur fédéré et la date d'expiration des informations d'identification.

Example Exemple de réponse

```
<AssumeRoleResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<AssumeRoleResult>
<SourceIdentity>DevUser123</SourceIdentity>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCR/oLxBA==
  </SessionToken>
  <SecretAccessKey>
   wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-07-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
<AssumedRoleUser>
  <Arn>arn:aws:sts::123456789012:assumed-role/demo/John</Arn>
  <AssumedRoleId>ARO123EXAMPLE123:John</AssumedRoleId>
</AssumedRoleUser>
<PackedPolicySize>8</PackedPolicySize>
</AssumeRoleResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</AssumeRoleResponse>
```

> **Note**
>
> Une AWS conversion compresse les politiques de session et les balises de session adoptées dans un format binaire compressé doté d'une limite distincte. Votre demande peut échouer pour cette limite, même si votre texte brut répond aux autres exigences. L'élément de réponse `PackedPolicySize` indique en pourcentage la proximité des stratégies et des balises de votre requête par rapport à la limite de taille supérieure.

AssumeRoleWithWebIdentity : fédération via un fournisseur d'identité basé sur le Web

L'opération d'API `AssumeRoleWithWebIdentity` renvoie un ensemble d'informations d'identification de sécurité temporaires pour les utilisateurs fédérés authentifiés par le biais d'un fournisseur d'identité public. Login with Amazon, Facebook, Google ou tout fournisseur d'identité compatible avec OpenID Connect (OIDC) sont des exemples de fournisseurs d'identité publics. Cette opération est utile pour créer des applications mobiles ou des applications Web basées sur le client qui nécessitent un accès à. AWS L'utilisation de cette opération signifie que vos utilisateurs n'ont pas besoin de leur propre identité AWS ou de leur identité IAM. Pour plus d'informations, consultez Fédération OIDC.

Au lieu d'appeler directement`AssumeRoleWithWebIdentity`, nous vous recommandons d'utiliser Amazon Cognito et le fournisseur d'informations d'identification Amazon Cognito avec les SDK pour AWS le développement mobile. Pour plus d'informations, consultez Authentification avec Amplify dans la documentation Amplify.

Si vous n'utilisez pas Amazon Cognito, vous appelez l'action `AssumeRoleWithWebIdentity` de AWS STS. Il s'agit d'un appel non signé, ce qui signifie que l'application n'a pas besoin d'accéder aux informations d'identification AWS pour effectuer l'appel. Lorsque vous effectuez cet appel, vous transmettez les informations suivantes :

- L'Amazon Resource Name (ARN) du rôle que l'application doit endosser. Si votre application permet aux utilisateurs de se connecter de plusieurs façons, vous devez définir plusieurs rôles, à raison d'un rôle par fournisseur d'identité. L'appel de `AssumeRoleWithWebIdentity` doit inclure l'ARN du rôle spécifique au fournisseur par le biais duquel l'utilisateur se connecte.
- Le jeton que l'IdP fourni à l'application une fois que celle-ci a authentifié l'utilisateur.
- Vous pouvez configurer votre fournisseur d'identité pour qu'il transmette des attributs à votre jeton en tant que balises de session.

- (Facultatif) La durée, qui indique la durée des informations d'identification temporaires. Utilisez le paramètre `DurationSeconds` pour spécifier la durée de la session du rôle entre 900 secondes (15 minutes) et la durée de session maximale pour le rôle. Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Affichage du paramètre de durée de session maximale pour un rôle. Si vous ne transmettez pas ce paramètre, les informations d'identification temporaires expirent au bout d'une heure. Le paramètre `DurationSeconds` de cette API est différent du paramètre HTTP `SessionDuration` que vous utilisez pour spécifier la durée de la session d'une console. Utilisez le paramètre HTTP `SessionDuration` de la demande au point de terminaison de fédération d'un jeton de connexion à la console. Pour plus d'informations, veuillez consulter Activation de l'accès à la AWS console par un courtier d'identité personnalisé.
- Nom de la session de rôle. Utilisez cette valeur de chaîne pour identifier la session lorsqu'un rôle est utilisé par différents principaux. Pour des raisons de sécurité, les administrateurs peuvent afficher ce champ dans les journaux AWS CloudTrail pour savoir qui a effectué une action dans AWS. Votre administrateur peut vous demander de fournir une valeur spécifique pour le nom de session lorsque vous endossez le rôle. Pour plus d'informations, consultez `sts:RoleSessionName`.
- (Facultatif) Identité source. Vous pouvez exiger des utilisateurs fédérés qu'ils spécifient une identité source lorsqu'ils endossent un rôle. Une fois l'identité source définie, la valeur ne peut pas être modifiée. Elle est présente dans la demande pour toutes les actions qui sont prises durant la session de rôle. La valeur d'identité source persiste entre les sessions de chaînage de rôles. Vous pouvez utiliser les informations d'identité de la source dans AWS CloudTrail les journaux pour déterminer qui a effectué des actions avec un rôle. Pour de plus amples informations sur l'utilisation de l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.
- (Facultatif) Stratégies de session en ligne ou gérées. Ces stratégies limitent les autorisations à partir de la stratégie basée sur l'identité du rôle qui sont attribuées à la session de rôle. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité du rôle et des politiques de session. Les politiques de session ne peuvent pas être utilisées pour accorder plus d'autorisations que celles autorisées par la politique basée sur l'identité du rôle endossé actuellement. Pour de plus amples informations sur les autorisations de session de rôle, veuillez consulter Politiques de session.

> **Note**
>
> Un appel à `AssumeRoleWithWebIdentity` n'est pas signé (chiffré). Par conséquent, vous devez uniquement inclure des politiques de session facultatives si la demande est

transmise via un intermédiaire approuvé. Dans ce cas, une personne peut modifier la politique afin de supprimer les restrictions.

Lorsque vous appelez`AssumeRoleWithWebIdentity`, AWS vérifie l'authenticité du jeton. Par exemple, selon le fournisseur, vous AWS pouvez passer un appel au fournisseur et inclure le jeton transmis par l'application. En supposant que le fournisseur d'identité valide le jeton, il vous AWS renvoie les informations suivantes :

- Un ensemble d'informations d'identification de sécurité temporaires. Il s'agit d'un ID de clé d'accès, d'une clé d'accès secrète et d'un jeton de session.
- L'ID de rôle et l'ARN du rôle endossé.
- Une valeur `SubjectFromWebIdentityToken` qui contient l'ID utilisateur unique.

Lorsque vous disposez des informations d'identification de sécurité temporaires, vous pouvez les utiliser pour effectuer des appels AWS d'API. Ce processus est identique à celui d'un appel d' AWS API avec des informations de sécurité à long terme. La différence est que vous devez inclure le jeton de session, ce qui permet à AWS de vérifier la validité des informations d'identification de sécurité temporaires.

Votre application met en cache les informations d'identification. Comme indiqué précédemment, par défaut, les informations d'identification expirent au bout d'une heure. Si vous n'utilisez pas l'CredentialsProvideropération AmazonSTS dans le AWS SDK, c'est à vous et à votre application de réappelerAssumeRoleWithWebIdentity. Appelez cette opération pour obtenir un nouvel ensemble d'informations d'identification de sécurité temporaires avant l'expiration des anciennes.

AssumeRoleWithSAML —fédération via un fournisseur d'identité d'entreprise compatible avec SAML 2.0

L'opération d'API `AssumeRoleWithSAML` renvoie un ensemble d'informations d'identification de sécurité temporaires pour les utilisateurs fédérés authentifiés par le système d'identité existant de votre organisation. Les utilisateurs doivent également utiliser SAML 2.0 (Security Assertion Markup Language) pour transmettre les informations d'authentification et d'autorisation à AWS. Cette opération d'API s'avère utile pour les organisations qui ont intégré leurs systèmes d'identité (comme Windows Active Directory ou OpenLDAP) à un logiciel capable de générer des assertions SAML. Cette intégration fournit des informations sur l'identité et les autorisations des utilisateurs (comme

Active Directory Federation Services ou Shibboleth). Pour plus d'informations, veuillez consulter Fédération SAML 2.0.

> **Note**
>
> Un appel à `AssumeRoleWithSAML` n'est pas signé (chiffré). Par conséquent, vous devez uniquement inclure des politiques de session facultatives si la demande est transmise via un intermédiaire approuvé. Dans ce cas, une personne peut modifier la politique afin de supprimer les restrictions.

Il s'agit d'un appel non signé, ce qui signifie que l'application n'a pas besoin d'accéder aux informations d'identification AWS pour effectuer l'appel. Lorsque vous effectuez cet appel, vous transmettez les informations suivantes :

- L'Amazon Resource Name (ARN) du rôle que l'application doit endosser.
- L'ARN du fournisseur SAML créé dans IAM qui décrit le fournisseur d'identité.
- L'assertion SAML, encodée en base64, qui a été fournie par le fournisseur d'identité SAML dans sa réponse d'authentification SAML à la demande de connexion de votre application.
- Vous pouvez configurer votre fournisseur d'identité pour qu'il transmette des attributs à votre assertion SAML en tant que balises de session.
- (Facultatif) La durée, qui indique la durée des informations d'identification temporaires. Utilisez le paramètre `DurationSeconds` pour spécifier la durée de la session du rôle entre 900 secondes (15 minutes) et la durée de session maximale pour le rôle. Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Affichage du paramètre de durée de session maximale pour un rôle. Si vous ne transmettez pas ce paramètre, les informations d'identification temporaires expirent au bout d'une heure. Le paramètre `DurationSeconds` de cette API est différent du paramètre HTTP `SessionDuration` que vous utilisez pour spécifier la durée de la session d'une console. Utilisez le paramètre HTTP `SessionDuration` de la demande au point de terminaison de fédération d'un jeton de connexion à la console. Pour plus d'informations, veuillez consulter Activation de l'accès à la AWS console par un courtier d'identité personnalisé.
- (Facultatif) Stratégies de session en ligne ou gérées. Ces stratégies limitent les autorisations à partir de la stratégie basée sur l'identité du rôle qui sont attribuées à la session de rôle. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité du rôle et des politiques de session. Les politiques de session ne peuvent pas être utilisées pour accorder plus d'autorisations que celles autorisées par la politique basée sur l'identité du rôle

endossé actuellement. Pour de plus amples informations sur les autorisations de session de rôle, veuillez consulter Politiques de session.

- Nom de la session de rôle. Utilisez cette valeur de chaîne pour identifier la session lorsqu'un rôle est utilisé par différents principaux. Pour des raisons de sécurité, les administrateurs peuvent afficher ce champ dans les journaux AWS CloudTrail pour savoir qui a effectué une action dans AWS. Votre administrateur peut vous demander de fournir une valeur spécifique pour le nom de session lorsque vous endossez le rôle. Pour plus d'informations, consultez `sts:RoleSessionName`.
- (Facultatif) Identité source. Vous pouvez exiger des utilisateurs fédérés qu'ils spécifient une identité source lorsqu'ils endossent un rôle. Une fois l'identité source définie, la valeur ne peut pas être modifiée. Elle est présente dans la demande pour toutes les actions qui sont prises durant la session de rôle. La valeur d'identité source persiste entre les sessions de chaînage de rôles. Vous pouvez utiliser les informations d'identité de la source dans AWS CloudTrail les journaux pour déterminer qui a effectué des actions avec un rôle. Pour de plus amples informations sur l'utilisation de l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

Lorsque vous appelez`AssumeRoleWithSAML`, AWS vérifie l'authenticité de l'assertion SAML. En supposant que le fournisseur d'identité valide l'assertion, il vous AWS renvoie les informations suivantes :

- Un ensemble d'informations d'identification de sécurité temporaires. Il s'agit d'un ID de clé d'accès, d'une clé d'accès secrète et d'un jeton de session.
- L'ID de rôle et l'ARN du rôle endossé.
- Une valeur `Audience` qui contient la valeur de l'attribut `Recipient` de l'élément `SubjectConfirmationData` de l'assertion SAML.
- Une valeur `Issuer` qui contient la valeur de l'élément `Issuer` de l'assertion SAML.
- `NameQualifier`Élément contenant une valeur de hachage créée à partir de la `Issuer` valeur, de l' Compte AWS ID et du nom convivial du fournisseur SAML. Lors de la combinaison avec l'élément `Subject`, ils identifient de manière unique l'utilisateur fédéré.
- Un élément `Subject` qui contient la valeur de l'élément `NameID` dans l'élément `Subject` de l'assertion SAML.
- Un élément `SubjectType` qui indique le format de l'élément `Subject`. La valeur peut être `persistent`, `transient` ou l'URI `Format` complet des éléments `Subject` et `NameID` utilisés

dans votre assertion SAML. Pour plus d'informations sur l'attribut `NameID` de l'élément `Format`, consultez Configurer les assertions SAML pour la réponse d'authentification.

Lorsque vous disposez des informations d'identification de sécurité temporaires, vous pouvez les utiliser pour effectuer des appels AWS d'API. Ce processus est identique à celui d'un appel d' AWS API avec des informations de sécurité à long terme. La différence est que vous devez inclure le jeton de session, ce qui permet à AWS de vérifier la validité des informations d'identification de sécurité temporaires.

Votre application met en cache les informations d'identification. Par défaut, les informations d'identification expirent au bout d'une heure. Si vous n'utilisez pas l'CredentialsProvideraction AmazonSTS dans le AWS SDK, c'est à vous et à votre application de rappeler`AssumeRoleWithSAML`. Appelez cette opération pour obtenir un nouvel ensemble d'informations d'identification de sécurité temporaires avant l'expiration des anciennes.

GetFederationToken : fédération via un broker d'identité personnalisé

L'opération d'API `GetFederationToken` retourne un ensemble d'informations d'identification de sécurité temporaires pour les utilisateurs fédérés. La différence entre cette API et `AssumeRole` réside dans le fait que le délai d'expiration par défaut est considérablement plus long (12 heures au lieu d'une heure). De plus, vous pouvez utiliser le paramètre `DurationSeconds` pour spécifier une durée pour que les informations d'identification de sécurité temporaires restent valides. Les informations d'identification obtenues sont valides pendant la durée spécifiée, comprise entre 900 secondes (15 minutes) et 129 600 secondes (36 heures). La période d'expiration plus longue peut contribuer à réduire le nombre d'appels, AWS car vous n'avez pas besoin de nouvelles informations d'identification aussi souvent.

Lorsque vous effectuez cette demande, vous utilisez les informations d'identification d'un utilisateur IAM spécifique. Les autorisations octroyées avec les informations d'identification de sécurité temporaires sont déterminées par les politiques de session transmises lors de l'appel de `GetFederationToken`. Les autorisations de session obtenues sont une combinaison des politiques d'utilisateur IAM et des politiques de session que vous transmettez. Les politiques de session ne peuvent pas être utilisées pour accorder plus d'autorisations que celles autorisées par la politique basée sur les identités de l'utilisateur IAM qui demande la fédération. Pour de plus amples informations sur les autorisations de session de rôle, veuillez consulter Politiques de session.

Lorsque vous utilisez les informations d'identification temporaires renvoyées par l'opération `GetFederationToken`, les étiquettes du principal de la session incluent les étiquettes de l'utilisateur

et les étiquettes de session transmises. Pour de plus amples informations sur les balises de session, veuillez consulter Transmission des balises de session AWS STS.

L'appel à `GetFederationToken` retourne des informations d'identification de sécurité temporaires constituées d'un jeton de session, d'une clé d'accès, d'une clé secrète et d'un délai d'expiration. Vous pouvez utiliser `GetFederationToken` si vous souhaiter gérer les autorisations au sein de l'organisation (par exemple, pour octroyer des autorisations à l'aide de l'application proxy).

L'exemple suivant illustre une demande et une réponse qui utilisent `GetFederationToken`. Cet exemple de demande fédère l'utilisateur appelant pour la durée spécifiée avec l'ARN de politique de session et les balises de session. La session obtenue est nommée `Jane-session`.

Example Exemple de demande

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetFederationToken
&Name=Jane-session
&PolicyArns.member.1.arn==arn%3Aaws%3Aiam%3A%3A123456789012%3Apolicy%2FRole1policy
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&AUTHPARAMS
```

L'ARN de politique illustrée dans l'exemple précédent inclut les ARN encodés en URL suivants :

`arn:aws:iam::123456789012:policy/Role1policy`

Notez également que le paramètre `&AUTHPARAMS` dans l'exemple est conçu comme un espace réservé pour les informations d'authentification. Il s'agit de la signature que vous devez inclure dans les demandes d'API AWS HTTP. Lors de la création de demandes d'API, nous vous recommandons d'utiliser les kits SDK AWS, car ceux-ci gèrent diverses tâches pour vous, notamment la signature des demandes. Si vous devez créer et signer des demandes d'API manuellement, consultez la section Signing AWS Requests By Using Signature Version 4 du Référence générale d'Amazon Web Servicespour savoir comment signer une demande.

Outre les informations d'identification de sécurité temporaires, la réponse inclut l'Amazon Resource Name (ARN) de l'utilisateur fédéré et la date d'expiration des informations d'identification.

Example Exemple de réponse

```
<GetFederationTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetFederationTokenResult>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCEXAMPLE==
  </SessionToken>
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-04-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE;</AccessKeyId>
</Credentials>
<FederatedUser>
  <Arn>arn:aws:sts::123456789012:federated-user/Jean</Arn>
  <FederatedUserId>123456789012:Jean</FederatedUserId>
</FederatedUser>
<PackedPolicySize>4</PackedPolicySize>
</GetFederationTokenResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</GetFederationTokenResponse>
```

> **Note**
>
> Une AWS conversion compresse les politiques de session et les balises de session adoptées dans un format binaire compressé doté d'une limite distincte. Votre demande peut échouer pour cette limite, même si votre texte brut répond aux autres exigences. L'élément de réponse `PackedPolicySize` indique en pourcentage la proximité des stratégies et des balises de votre requête par rapport à la limite de taille supérieure.

AWS recommande d'accorder des autorisations au niveau des ressources (par exemple, si vous attachez une politique basée sur les ressources à un compartiment Amazon S3), vous pouvez omettre le paramètre. `Policy` Toutefois, si vous n'incluez pas de politique pour un utilisateur fédéré,

les informations d'identification de sécurité temporaires ne lui accordent aucune autorisation. Dans ce cas, vous devez utiliser des politiques de ressources pour accorder à l'utilisateur fédéré l'accès à vos ressources AWS .

Par exemple, supposons que votre Compte AWS numéro est 111122223333 et que vous souhaitez autoriser Susan à accéder à un compartiment Amazon S3. Les informations d'identification de sécurité temporaires de Susan ne comprennent pas de politique pour ce compartiment. Dans ce cas, vous devez vous assurer que le compartiment est doté d'une politique avec un ARN correspondant à celui de Susan, par exemple `arn:aws:sts::111122223333:federated-user/Susan`.

GetSessionToken : informations d'identification temporaires pour les utilisateurs dans des environnements non de confiance

L'opération d'API `GetSessionToken` retourne un ensemble d'informations d'identification de sécurité temporaires pour un utilisateur IAM existant. Cela est utile pour renforcer la sécurité, par exemple pour autoriser les AWS demandes uniquement lorsque la MFA est activée pour l'utilisateur IAM. Dans la mesure où les informations d'identification sont temporaires, elles offrent une sécurité améliorée lorsqu'un utilisateur IAM accède à vos ressources par le biais d'un environnement moins sécurisé. Les exemples d'environnements moins sécurisés incluent un périphérique mobile ou un navigateur web. Pour plus d'informations, consultez Demande d'informations d'identification temporaires de sécurité ou consultez GetSessionTokenle Guide de référence des AWS Security Token Service API.

Par défaut, les informations d'identification de sécurité temporaires d'un utilisateur IAM restent valides pendant 12 heures maximum. Mais vous pouvez demander un durée de validité allant de 15 minutes à 36 heures à l'aide du paramètre `DurationSeconds`. Pour des raisons de sécurité, la durée d'un jeton Utilisateur racine d'un compte AWS est limitée à une heure.

`GetSessionToken` retourne des informations d'identification de sécurité temporaires constituées d'un jeton de session, d'un ID de clé d'accès et d'une clé d'accès secrète. L'exemple suivant illustre une demande et une réponse qui utilisent `GetSessionToken`. La réponse inclut également l'heure d'expiration des informations d'identification de sécurité temporaires.

Example Exemple de demande

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=1800
```

```
&AUTHPARAMS
```

Le paramètre AUTHPARAMS de l'exemple est un espace réservé pour votre signature. Une signature est l'information d'authentification que vous devez inclure dans les demandes d'API AWS HTTP. Lors de la création de demandes d'API, nous vous recommandons d'utiliser les kits SDK AWS, car ceux-ci gèrent diverses tâches pour vous, notamment la signature des demandes. Si vous devez créer et signer des demandes d'API manuellement, consultez la section Signing AWS Requests By Using Signature Version 4 du Référence générale d'Amazon Web Servicespour savoir comment signer une demande.

Example Exemple de réponse

```
<GetSessionTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetSessionTokenResult>
<Credentials>
  <SessionToken>
   AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT+FvwqnKwRcOIfrRh3c/L
   To6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/IvU1dYUg2RVAJBanLiHb4IgRmpRV3z
   rkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/AXlzBBko7b15fjrBs2+cTQtp
   Z3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE
  </SessionToken>
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2011-07-11T19:55:29.611Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
</GetSessionTokenResult>
<ResponseMetadata>
<RequestId>58c5dbae-abef-11e0-8cfe-09039844ac7d</RequestId>
</ResponseMetadata>
</GetSessionTokenResponse>
```

Facultativement, la `GetSessionToken` demande peut inclure `SerialNumber` des `TokenCode` valeurs pour la vérification de l' AWS authentification multifactorielle (MFA). Si les valeurs fournies sont valides, AWS STS fournit des informations d'identification de sécurité temporaires qui incluent l'état de l'authentification MFA. Les informations d'identification de sécurité temporaires peuvent ensuite être utilisées pour accéder aux opérations d'API ou aux AWS sites Web protégés par le MFA tant que l'authentification MFA est valide.

L'exemple suivant illustre une demande `GetSessionToken` qui comprend un code de vérification MFA et un numéro de série du dispositif.

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=7200
&SerialNumber=YourMFADeviceSerialNumber
&TokenCode=123456
&AUTHPARAMS
```

> **Note**
>
> L'appel AWS STS peut être dirigé vers le point de terminaison mondial ou vers l'un des points de terminaison régionaux pour lesquels vous activez votre Compte AWS. Pour plus d'informations, veuillez consulter la section AWS STS de Régions et points de terminaison. Le paramètre `AUTHPARAMS` de l'exemple est un espace réservé pour votre signature. Une signature est l'information d'authentification que vous devez inclure dans les demandes d'API AWS HTTP. Lors de la création de demandes d'API, nous vous recommandons d'utiliser les kits SDK AWS, car ceux-ci gèrent diverses tâches pour vous, notamment la signature des demandes. Si vous devez créer et signer des demandes d'API manuellement, consultez la section Signature des AWS demandes à l'aide de la version 4 du Référence générale d'Amazon Web Servicespour savoir comment signer une demande.

Comparaison des opérations AWS STS d'API

Le tableau suivant compare les fonctionnalités des opérations d'API AWS STS qui renvoient des informations d'identification de sécurité temporaires. Pour en savoir plus sur les différentes méthodes disponibles pour demander des identifiants de sécurité temporaires en endossant un rôle, consultez Utilisation de rôles IAM. Pour en savoir plus sur les différentes opérations d' AWS STS API qui vous permettent de transmettre des balises de session, consultezTransmission des balises de session AWS STS.

Comparaison des options d'API

AWS STS API	Qui peut appeler	Durée de vie des informations d'identification (min | max | par défaut)	Prise en charge de l'authentification MFA[1]	Prise en charge de la politique de session[2]	Restrictions applicables aux informations d'identification temporaires générées
AssumeRole	Utilisateur IAM ou rôle IAM disposant d'informations d'identification de sécurité temporaires existantes	15 min | Durée de session maximale | 1 h	Oui	Oui	Impossible d'appeler `GetFederationToken` ou `GetSessionToken` .
AssumeRoleWithSAML	N'importe quel utilisateur ; le principal doit transmettre une réponse d'authentification SAML qui spécifie l'authentification par un fournisseur d'identité reconnu	15 min | Durée de session maximale | 1 h	Non	Oui	Impossible d'appeler `GetFederationToken` ou `GetSessionToken` .
AssumeRoleWithWebIdentity	Tout utilisateur ; l'appelant doit transmettre un jeton JWT conforme à l'OIDC qui indique	15 min | Durée de session maximale | 1 h	Non	Oui	Impossible d'appeler `GetFederationToken` ou `GetSessionToken` .

AWS STS API	Qui peut appeler	Durée de vie des informations d'identification (min | max | par défaut)	Prise en charge de l'authentification MFA[1]	Prise en charge de la politique de session[2]	Restrictions applicables aux informations d'identification temporaires générées
	l'authentification auprès d'un fournisseur d'identité connu				
GetFederationToken	Utilisateur IAM ou Utilisateur racine d'un compte AWS	Utilisateur IAM : 15 min | 36 h | 12 h Utilisateur racine : 15 min | 1 heure | 1 heure	Non	Oui	Impossible d'appeler des opérations IAM à l'aide de l'AWS API AWS CLI or. Cette limitation ne s'applique pas aux sessions de console. Impossible d'appeler AWS STS les opérations sauf `GetCallerIdentity` .[4] Connexion avec authentification unique (SSO) à la console autorisée.[5]

AWS STS API	Qui peut appeler	Durée de vie des informations d'identification (min | max | par défaut)	Prise en charge de l'authentification MFA[1]	Prise en charge de la politique de session[2]	Restrictions applicables aux informations d'identification temporaires générées
GetSessionToken	Utilisateur IAM ou Utilisateur racine d'un compte AWS	Utilisateur IAM : 15 min | 36 h | 12 h Utilisateur racine : 15 min | 1 heure | 1 heure	Oui	Non	Impossible d'appeler les opérations d'API IAM si les informations MFA ne figurent pas dans la demande. Impossible d'appeler les opérations d' AWS STS API sauf `AssumeRole` ou`GetCallerIdentity` . Connexion avec authentification unique (SSO) à la console non autorisée.[6]

[1] Prise en charge de l'authentification MFA. Vous pouvez inclure des informations sur un dispositif d'authentification multifactorielle (MFA) lorsque vous appelez AssumeRole les opérations GetSessionToken et API. De cette façon, les informations d'identification de sécurité temporaires obtenues lors l'appel d'API peuvent uniquement être utilisées par des utilisateurs authentifiés à l'aide d'un dispositif MFA. Pour plus d'informations, veuillez consulter Configuration de l'accès aux API protégé par MFA.

[2] Prise en charge de la politique de session. Les politiques de session sont des politiques que vous passez en tant que paramètre lorsque vous créez par programmation une session temporaire pour un rôle ou un utilisateur fédéré. Cette politique limite les autorisations à partir de la politique basée sur l'identité du rôle ou de l'utilisateur qui sont attribuées à la session. Les autorisations de la session

obtenues sont une combinaison des stratégies basées sur l'identité de l'entité et des stratégies de session. Les politiques de session ne peuvent pas être utilisées pour accorder plus d'autorisations que celles autorisées par la politique basée sur l'identité du rôle endossé actuellement. Pour de plus amples informations sur les autorisations de session de rôle, veuillez consulter Politiques de session.

[3] Paramètre de durée de session maximale. Utilisez le paramètre `DurationSeconds` pour spécifier la durée de la session de votre rôle entre 900 secondes (15 minutes) et la durée de session maximale pour le rôle. Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Affichage du paramètre de durée de session maximale pour un rôle.

[4] GetCallerIdentity. Aucune autorisation n'est requise pour effectuer cette opération. Si un administrateur ajoute une politique à votre utilisateur ou rôle IAM qui refuse explicitement l'accès à l'action `sts:GetCallerIdentity`, vous pouvez toujours effectuer cette opération. Les autorisations ne sont pas requises, car les mêmes informations sont renvoyées lorsqu'un utilisateur ou un rôle IAM se voit refuser l'accès. Pour afficher un exemple de réponse, consultez Je ne suis pas autorisé à effectuer : iam : MFAdevice DeleteVirtual.

[5] Connexion avec authentification unique (SSO) à la console. Pour prendre en charge l'authentification unique, vous AWS pouvez appeler un point de terminaison de fédération (`https://signin.aws.amazon.com/federation`) et transmettre des informations d'identification de sécurité temporaires. Le point de terminaison retourne un jeton que vous pouvez utiliser pour créer une URL qui connecte directement l'utilisateur à la console, sans avoir besoin d'un mot de passe. Pour plus d'informations, consultez Permettre aux utilisateurs fédérés SAML 2.0 d'accéder au AWS Management Console la section Comment activer l'accès multicompte à la console de AWS gestion dans le blog sur la AWS sécurité.

3 Une fois que vous avez récupéré vos informations d'identification temporaires, vous ne pouvez pas y accéder AWS Management Console en les transmettant au point de terminaison d'authentification unique de la fédération. Pour plus d'informations, voir Activation de l'accès à la AWS console par un courtier d'identité personnalisé.

Utilisation d'informations d'identification temporaires avec des ressources AWS

Vous pouvez utiliser des informations d'identification de sécurité temporaires pour effectuer des demandes programmatiques de AWS ressources à l'aide de l' AWS API AWS CLI or (à l'aide des AWS SDK). Les informations d'identification temporaires fournissent les mêmes autorisations que les informations d'identification de sécurité à long terme, telles que les informations d'identification de l'utilisateur IAM. Toutefois, il existe quelques différences :

- Lorsque vous passez un appel à l'aide d'informations d'identification de sécurité temporaires, l'appel doit inclure un jeton de session, qui est renvoyé avec ces informations d'identification temporaires. AWS utilise le jeton de session pour valider les informations d'identification de sécurité temporaires.
- Les informations d'identification temporaires arrivent à expiration après un intervalle spécifique. Une fois les informations d'identification temporaires arrivées à expiration, tous les appels que vous effectuez avec elles échoueront. Vous devez donc générer un nouvel ensemble d'informations d'identification temporaires. Les informations d'identification temporaires ne peuvent pas être étendues ou actualisées au-delà de l'intervalle spécifié d'origine.
- Lorsque vous utilisez des informations d'identification temporaires pour effectuer une demande, votre principal peut inclure un ensemble d'étiquettes. Ces balises proviennent de balises de session et de balises attachées au rôle que vous endossez. Pour de plus amples informations sur les balises de session, veuillez consulter Transmission des balises de session AWS STS.

Si vous utilisez les AWS SDK, le AWS Command Line Interface(AWS CLI) ou les outils pour Windows PowerShell, la manière d'obtenir et d'utiliser les informations d'identification de sécurité temporaires varie selon le contexte. Si vous exécutez du code ou AWS CLI des PowerShell commandes Tools for Windows dans une instance EC2, vous pouvez tirer parti des rôles pour Amazon EC2. Sinon, vous pouvez appeler une API AWS STS pour obtenir les informations d'identification temporaires, puis les utiliser explicitement pour effectuer des appels aux services AWS .

> **Note**
>
> Vous pouvez utiliser AWS Security Token Service (AWS STS) pour créer et fournir à des utilisateurs de confiance des informations d'identification de sécurité temporaires qui peuvent contrôler l'accès à vos AWS ressources. Pour plus d'informations sur AWS STS, voirInformations d'identification de sécurité temporaires dans IAM. AWS STS est un service global dont le point de terminaison par défaut est situé à`https://sts.amazonaws.com`. Ce point de terminaison se trouve dans la région USA Est (Virginie du Nord), bien que les informations d'identification que vous obtenez de ce point de terminaison et d'autres soient valides dans le monde entier. Ces informations d'identification fonctionnent avec les services et les ressources dans n'importe quelle région. Vous pouvez également choisir d'effectuer des appels d' AWS STS API vers des points de terminaison situés dans l'une des régions prises en charge. Cela permet de réduire la latence en effectuant les demandes depuis les serveurs situés dans une région géographiquement plus proche de vous. Quelle que soit la

région d'où proviennent vos informations d'identification, elles fonctionnent dans le monde entier. Pour plus d'informations, veuillez consulter Gérer AWS STS dans un Région AWS.

Table des matières

- Utilisation d'informations d'identification temporaires dans les instances Amazon EC2
- Utilisation d'informations d'identification de sécurité temporaires avec les kits SDK AWS
- Utilisation d'informations d'identification de sécurité temporaires avec la AWS CLI
- Utilisation d'informations d'identification de sécurité temporaires avec des opérations d'API
- En savoir plus

Utilisation d'informations d'identification temporaires dans les instances Amazon EC2

Si vous souhaitez exécuter des AWS CLI commandes ou du code dans une instance EC2, il est recommandé d'utiliser des rôles pour Amazon EC2 pour obtenir des informations d'identification. Vous créez un rôle IAM qui spécifie les autorisations que vous souhaitez accorder aux applications qui s'exécutent sur les instances EC2. Lorsque vous lancez l'instance, vous associez le rôle à cette instance.

Les PowerShell commandes Applications et Outils pour Windows qui s'exécutent sur l'instance peuvent ensuite obtenir des informations d'identification de sécurité temporaires automatiques à partir des métadonnées de l'instance. AWS CLI Vous n'avez pas besoin d'obtenir explicitement les informations d'identification de sécurité temporaires. Les AWS SDK et les AWS CLI outils pour Windows obtiennent PowerShell automatiquement les informations d'identification du service de métadonnées d'instance EC2 (IMDS) et les utilisent. Les informations d'identification temporaires disposent d'autorisations que vous définissez pour le rôle associé à l'instance.

Pour plus d'informations et d'exemples, consultez ce qui suit :

- Utilisation des rôles IAM pour accorder l'accès aux AWS ressources sur Amazon Elastic Compute Cloud — AWS SDK for Java
- Octroi d'accès à l'aide d'un rôle IAM — AWS SDK for .NET
- Création d'un rôle — AWS SDK for Ruby

Utilisation d'informations d'identification de sécurité temporaires avec les kits SDK AWS

Pour utiliser des informations d'identification de sécurité temporaires dans le code, vous appelez par programmation une AWS STS API similaire `AssumeRole` et vous extrayez les informations d'identification et le jeton de session qui en résultent. Vous utilisez ensuite ces valeurs comme informations d'identification pour les appels suivants à AWS. L'exemple suivant montre un pseudocode expliquant comment utiliser les informations d'identification de sécurité temporaires si vous utilisez un AWS SDK :

```
assumeRoleResult = AssumeRole(role-arn);
tempCredentials = new SessionAWSCredentials(
   assumeRoleResult.AccessKeyId,
   assumeRoleResult.SecretAccessKey,
   assumeRoleResult.SessionToken);
s3Request = CreateAmazonS3Client(tempCredentials);
```

Pour un exemple écrit en Python (en utilisant l'AWS SDK for Python (Boto)), veuillez consulter Passage à un rôle IAM (AWS API). Cet exemple montre comment appeler `AssumeRole` pour obtenir des informations d'identification de sécurité temporaires, puis utiliser ces informations d'identification pour appeler Amazon S3.

Pour plus d'informations sur la façon d'appeler `AssumeRole`, `GetFederationToken` et d'autres opérations d'API, consultez la Référence sur l'API AWS Security Token Service. Pour plus d'informations sur l'obtention des informations d'identification de sécurité temporaires et du jeton de session à partir des résultats, consultez la documentation du kit SDK que vous utilisez. Vous trouverez la documentation de tous les AWS SDK sur la page de AWS documentation principale, dans la section SDK et boîtes à outils.

Vous devez vérifier que vous obtenez un nouvel ensemble d'informations d'identification avant que les anciennes arrivent à expiration. Dans certains kits SDK, vous pouvez utiliser un fournisseur qui gère le processus d'actualisation des informations d'identification pour vous. Consultez la documentation du kit SDK que vous utilisez.

Utilisation d'informations d'identification de sécurité temporaires avec la AWS CLI

Vous pouvez utiliser les informations d'identification de sécurité temporaires avec l'interface AWS CLI. Cela peut vous être utile lors de test de politiques.

À l'aide de l'AWS CLI, vous pouvez appeler une API AWS STS telle que `AssumeRole` ou `GetFederationToken`, puis capturer la sortie obtenue. L'exemple suivant illustre un appel à `AssumeRole` qui envoie la sortie dans un fichier. Dans l'exemple, le `profile` paramètre est supposé être un profil dans le fichier AWS CLI de configuration. Il est également supposé référencer les informations d'identification d'un utilisateur IAM qui a les autorisations d'endosser le rôle.

```
aws sts assume-role --role-arn arn:aws:iam::123456789012:role/role-name --role-session-
name "RoleSession1" --profile IAM-user-name > assume-role-output.txt
```

Une fois l'exécution de la commande terminée, vous pouvez extraire l'ID de clé d'accès, la clé d'accès secrète et le jeton de session de la variable. Vous pouvez le faire manuellement ou à l'aide d'un script. Vous pouvez ensuite affecter ces valeurs aux variables d'environnement.

Lorsque vous exécutez des AWS CLI commandes, les AWS CLI informations d'identification sont recherchées dans un ordre spécifique, d'abord dans les variables d'environnement, puis dans le fichier de configuration. Par conséquent, une fois que vous avez placé les informations d'identification temporaires dans les variables d'environnement, elles AWS CLI utilisent ces informations d'identification par défaut. (Si vous spécifiez un `profile` paramètre dans la commande, les variables d'environnement AWS CLI sont ignorées. Ils apparaissent plutôt AWS CLI dans le fichier de configuration, qui vous permet de remplacer les informations d'identification des variables d'environnement si nécessaire.)

L'exemple suivant montre comment définir les variables d'environnement pour les informations d'identification de sécurité temporaires, puis appeler une AWS CLI commande. Comme aucun `profile` paramètre n'est inclus dans la AWS CLI commande, AWS CLI elle recherche d'abord les informations d'identification dans les variables d'environnement et utilise donc les informations d'identification temporaires.

Linux

```
$ export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of session token>
$ aws ec2 describe-instances --region us-west-1
```

Windows

```
C:\> SET AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
```

```
C:\> SET AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
C:\> SET AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of token>
C:\> aws ec2 describe-instances --region us-west-1
```

Utilisation d'informations d'identification de sécurité temporaires avec des opérations d'API

Si vous envoyez des demandes d'API HTTPS directes à AWS, vous pouvez signer ces demandes avec les informations d'identification de sécurité temporaires que vous obtenez du AWS Security Token Service (AWS STS). Pour ce faire, vous utilisez l'ID de clé d'accès et la clé d'accès secrète que vous recevez AWS STS. Vous utilisez l'ID de clé d'accès et la clé d'accès secrète de la même façon que vous utiliseriez des informations d'identification à long terme pour signer une demande. Vous ajoutez également à votre demande d'API le jeton de session que vous recevez AWS STS. Vous ajoutez le jeton de session à un en-tête HTTP ou à un paramètre de chaîne de requête appelé `X-Amz-Security-Token`. Vous ajoutez le jeton de session à l'en-tête HTTP ou au paramètre de chaîne de requête, mais pas les deux. Pour plus d'informations sur la signature des demandes d'API HTTPS, consultez la section Signature des demandes d' AWS API dans le Références générales AWS.

En savoir plus

Pour plus d'informations sur l'utilisation AWS STS avec d'autres AWS services, consultez les liens suivants :

- Amazon S3. Veuillez consulter Making Requests Using IAM User Temporary Credentials (Envoyer des demandes au moyen d'informations d'identification temporaires de l'utilisateur IAM) ou Making Requests Using Federated User Temporary Credentials (Envoyer des demandes au moyen d'informations d'identification temporaires de l'utilisateur fédéré) dans le Guide de l'utilisateur Amazon Simple Storage Service.
- Amazon SNS. Consultez la section Utilisation de politiques basées sur l'identité avec Amazon SNS dans le manuel du développeur Amazon Simple Notification Service.
- Amazon SQS. Consultez la section Gestion des identités et des accès dans Amazon SQS dans le manuel Amazon Simple Queue Service Developer Guide.
- Amazon SimpleDB. Veuillez consulter Using Temporary Security Credentials (Utilisation d'informations d'identification de sécurité temporaires) dans le Manuel du développeur Amazon SimpleDB.

Contrôle des autorisations affectées aux informations d'identification de sécurité temporaires

Vous pouvez utiliser AWS Security Token Service (AWS STS) pour créer et fournir à des utilisateurs de confiance des informations d'identification de sécurité temporaires qui peuvent contrôler l'accès à vos AWS ressources. Pour plus d'informations sur AWS STS, voirInformations d'identification de sécurité temporaires dans IAM. Lorsqu' AWS STS émet des informations d'identification de sécurité temporaires, celles-ci sont valides jusqu'au délai d'expiration spécifié et elles ne peuvent pas être annulées. Toutefois, les autorisations octroyées aux informations d'identification de sécurité temporaires sont évaluées chaque fois qu'une demande est effectuée à l'aide de ces informations. Il est donc possible de révoquer les informations d'identification en modifiant leurs droits d'accès après leur émission.

Les rubriques suivantes supposent que vous avez une connaissance pratique des AWS autorisations et des politiques. Pour plus d'informations sur ces rubriques, consultez Gestion de l'accès aux AWS ressources.

Rubriques

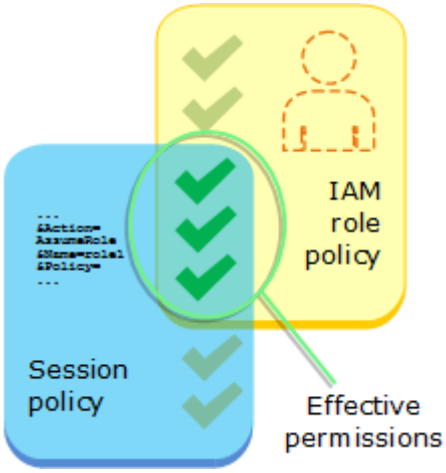
- Autorisations pour AssumeRole, AssumeRoleWith SAML et AssumeRoleWithWebIdentity
- Surveiller et contrôler les actions prises avec les rôles endossés
- Autorisations pour GetFederationToken
- Autorisations pour GetSessionToken
- Désactivation des autorisations affectées aux informations d'identification de sécurité temporaires
- Octroi d'autorisations pour créer des informations d'identification de sécurité temporaires
- Octroi d'autorisations pour utiliser des sessions de console sensibles à l'identité

Autorisations pour AssumeRole, AssumeRoleWith SAML et AssumeRoleWithWebIdentity

La politique d'autorisations du rôle endossé actuellement déterminé les autorisations octroyées aux informations d'identification de sécurité temporaires renvoyées par `AssumeRole`, `AssumeRoleWithSAML` et `AssumeRoleWithWebIdentity`. Vous définissez ces autorisations lors de la définition ou de la mise à jour du rôle.

Le cas échéant, vous pouvez transmettre des politiques de session en ligne ou gérées en tant que paramètre des opérations d'API `AssumeRole`, `AssumeRoleWithSAML` ou

`AssumeRoleWithWebIdentity`. Les politiques de session limitent les autorisations pour la session d'informations d'identification temporaires du rôle. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité du rôle et des politiques de session. Vous pouvez utiliser les informations d'identification temporaires du rôle lors des appels d' AWS API suivants pour accéder aux ressources du compte propriétaire du rôle. Vous ne pouvez pas utiliser les politiques de session pour accorder plus d'autorisations que celles autorisées par la politique basée sur l'identité du rôle endossé actuellement. Pour en savoir plus sur la façon dont AWS détermine les autorisations effectives d'un rôle, consultez Logique d'évaluation de politiques.



Les politiques associées aux informations d'identification à l'origine de l'appel ne `AssumeRole` sont pas évaluées AWS lors de la prise de la décision d' « autoriser » ou de « refuser » l'autorisation. L'utilisateur renonce temporairement à ses autorisations d'origine en faveur de celles affectées au rôle endossé. Dans le cas des opérations `AssumeRoleWithSAML` et de l'`AssumeRoleWithWebIdentity`API, il n'y a aucune politique à évaluer car l'appelant de l'API n'est pas une AWS identité.

Exemple : attribution d'autorisations à l'aide de AssumeRole

Vous pouvez utiliser l'opération d'API `AssumeRole` avec différents types de politiques. Voici quelques exemples.

Politique d'autorisations de rôle

Dans cet exemple, vous appelez l'opération d'API `AssumeRole` sans spécifier la politique de session dans le paramètre `Policy` facultatif. Les autorisations affectées aux informations d'identification temporaires sont déterminées par la politique d'autorisations du rôle endossé. L'exemple de politique d'autorisations suivant accorde au rôle l'autorisation de répertorier tous les objets contenus dans un compartiment S3 nommé `productionapp`. Il permet également au rôle d'obtenir, de placer et de supprimer des objets dans ce compartiment.

Example Exemple de politique d'autorisations de rôle

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Politique de session transmise en tant que paramètre

Imaginons que vous souhaitiez autoriser un utilisateur à endosser le même rôle que dans l'exemple suivant. Mais, dans ce cas, vous voulez que la session de rôle ait uniquement l'autorisation d'obtenir et de placer des objets dans le compartiment `productionapp` S3. Vous ne souhaitez pas l'autoriser à supprimer des objets. Pour cela, vous pouvez créer un rôle et spécifier les autorisations souhaitées dans la politique d'autorisations du rôle. Vous pouvez également appeler l'API `AssumeRole` et inclure des politiques de session dans le paramètre `Policy` facultatif dans le cadre de l'opération d'API. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité du rôle et des politiques de session. Les politiques de session ne peuvent pas être utilisées pour accorder plus d'autorisations que celles autorisées par la politique basée sur l'identité du rôle endossé actuellement. Pour de plus amples informations sur les autorisations de session de rôle, veuillez consulter Politiques de session.

Une fois que vous avez récupéré les informations d'identification temporaires de la nouvelle session, vous pouvez les transmettre à l'utilisateur auquel vous voulez donner ces autorisations.

Par exemple, imaginons que la politique suivante soit transmise en tant que paramètre de l'appel d'API. La personne qui utilise la session dispose d'autorisations pour effectuer uniquement les actions suivantes :

- Répertorier tous les objets du compartiment `productionapp`.
- Obtenir et placer des objets dans le compartiment `productionapp`.

Dans la session suivante, l'autorisation `s3:DeleteObject` est exclue du filtre et la session endossée n'obtient pas l'autorisation `s3:DeleteObject`. La politique définit les autorisations maximales pour la session de rôle afin qu'elle remplace les politiques d'autorisations existantes sur le rôle.

Example Exemple de politique de session transmise avec l'appel d'API **`AssumeRole`**

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Politique basée sur une ressource

Certaines AWS ressources prennent en charge les politiques basées sur les ressources, et ces politiques fournissent un autre mécanisme pour définir les autorisations qui affectent les informations d'identification de sécurité temporaires. Seules quelques ressources, comme les compartiments Amazon S3, les rubriques Amazon SNS et les files d'attente Amazon SQS, prennent en charge les

politiques basées sur des ressources. L'exemple suivant développe les exemples précédents, à l'aide d'un compartiment S3 nommé `productionapp`. La politique suivante est attachée au compartiment.

Lorsque vous attachez la politique basée sur les ressources suivante au compartiment `productionapp`, tous les utilisateurs se voient refuser l'autorisation de supprimer des objets du compartiment. (Voir l'élément `Principal` dans la politique.) Cela inclut tous les utilisateurs du rôle endossé, même si la politique d'autorisations du rôle accorde l'autorisation `DeleteObject`. Une instruction `Deny` explicite a toujours priorité sur une instruction `Allow`.

Example Exemple de politique de compartiment

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Principal": {"AWS": "*"},
    "Effect": "Deny",
    "Action": "s3:DeleteObject",
    "Resource": "arn:aws:s3:::productionapp/*"
  }
}
```

Pour plus d'informations sur la façon dont plusieurs types de politiques sont combinés et évalués par AWS, voirLogique d'évaluation de politiques.

Surveiller et contrôler les actions prises avec les rôles endossés

Un rôle IAM est un objet dans IAM auquel sont affectées des autorisations. Lorsque vous assumez ce rôle en utilisant une identité IAM ou une identité extérieure AWS, vous recevez une session avec les autorisations attribuées au rôle.

Lorsque vous effectuez des actions dans AWS, les informations relatives à votre session peuvent être enregistrées AWS CloudTrail pour que l'administrateur de votre compte puisse les surveiller. Les administrateurs peuvent configurer les rôles de sorte à exiger des identités qu'elles transmettent une chaîne personnalisée identifiant la personne ou l'application qui effectue des actions dans AWS. Ces informations d'identité sont stockées en tant qu'identité source dans AWS CloudTrail. Lorsque l'administrateur examine l'activité dans CloudTrail, il peut consulter les informations d'identité de la source afin de déterminer qui ou quoi a effectué des actions dans le cadre de sessions à rôle assumé.

Une fois qu'une identité source est définie, elle est présente dans les demandes concernant toute AWS action entreprise au cours de la session de rôle. La valeur définie persiste lorsqu'un rôle est

utilisé pour assumer un autre rôle via l' AWS API AWS CLI or, ce que l'on appelle le chaînage de rôles. La valeur définie ne peut pas être modifiée durant la session de rôle. Les administrateurs peuvent configurer des autorisations granulaires en fonction de la présence ou de la valeur de l'identité source afin de mieux contrôler les AWS actions entreprises avec des rôles partagés. Vous pouvez décider si l'attribut d'identité source peut être utilisé, s'il est requis ; et quelle valeur peut être utilisée.

L'utilisation de l'identité source est radicalement différente de celle du nom de session de rôle et des balises de session. La valeur d'identité source ne peut pas être modifiée une fois qu'elle est définie, et elle persiste pour toutes les actions supplémentaires qui sont prises durant la session de rôle. Voici comment utiliser les balises de session et le nom de session de rôle :

- Session tags (Balises de session) : vous pouvez transmettre des balises de session lorsque vous endossez un rôle ou fédérez un utilisateur. Les balises de session sont présentes lorsqu'un rôle est endossé. Vous pouvez définir des politiques qui utilisent des clés de condition de balise pour accorder des autorisations à vos principaux en fonction de leurs balises. Vous pouvez ensuite l'utiliser CloudTrail pour afficher les demandes faites pour assumer des rôles ou fédérer des utilisateurs. Pour en savoir plus sur les balises de session, veuillez consulter Transmission des balises de session AWS STS.
- Role session name (Nom de la session de rôle) : vous pouvez utiliser la clé de condition `sts:RoleSessionName` dans une politique d'approbation de rôle pour exiger que vos utilisateurs fournissent un nom de session spécifique lorsqu'ils endossent un rôle. Le nom de session de rôle peut servir à différencier les sessions de rôle lorsqu'un rôle est utilisé par différents principaux. Pour en savoir plus sur le nom de session de rôle, consultez sts : RoleSessionName.

Nous vous recommandons d'utiliser l'identité source pour contrôler l'identité qui endosse un rôle. L'identité de la source est également utile pour les CloudTrail journaux de minage afin de déterminer qui a utilisé le rôle pour effectuer des actions.

Rubriques

- Configuration d'utilisation de l'identité source
- Ce qu'il faut savoir sur l'identité source
- Autorisations requises pour définir l'identité source
- Spécification d'une identité source lorsqu'un rôle est endossé
- Utilisation de l'identité source avec AssumeRole

- Utilisation de l'identité source avec AssumeRoleWith SAML
- Utilisation de l'identité source avec AssumeRoleWithWebIdentity
- Contrôle de l'accès au moyen des informations d'identité source
- Affichage de l'identité de la source dans CloudTrail

Configuration d'utilisation de l'identité source

Votre configuration d'utilisation de l'identité source dépend de la méthode employée lorsque vos rôles sont endossés. Par exemple, vos utilisateurs IAM peuvent endosser des rôles directement via l'opération `AssumeRole`. Si vous possédez des identités d'entreprise, également appelées identités du personnel, elles peuvent accéder à vos AWS ressources en utilisant`AssumeRoleWithSAML`. Si des utilisateurs finaux accèdent à vos applications mobiles ou Web, ils peuvent le faire en utilisant `AssumeRoleWithWebIdentity`. La présentation du flux de haut niveau qui suit vous aidera à comprendre comment effectuer une configuration pour utiliser les informations d'identité source dans votre environnement existant.

1. Configurer les utilisateurs et les rôles test : en utilisant un environnement de préproduction, configurez les utilisateurs et les rôles test, et configurez leurs politiques afin d'autoriser la définition d'une identité source.

   Si vous utilisez un fournisseur d'identité (IdP) pour vos identités fédérées, configurez-le de sorte à transmettre un attribut utilisateur de votre choix pour l'identité source dans l'assertion ou le jeton.
2. Assumer le rôle : testez le fait d'endosser des rôles et de transmettre une identité source avec les utilisateurs et les rôles que vous configurez pour le test.
3. Révision CloudTrail : passez en revue les informations d'identité source de vos rôles de test dans vos CloudTrail journaux.
4. Entraîner vos utilisateurs : après avoir effectué des tests dans votre environnement de préproduction, vérifiez que vos utilisateurs savent parfaitement comment transmettre les informations d'identité source, si nécessaire. Définissez une date limite à laquelle vous exigerez de vos utilisateurs qu'ils fournissent une identité source dans votre environnement de production.
5. Configurer des politiques de production : configurez des politiques pour votre environnement de production, puis ajoutez-les à vos utilisateurs et rôles de production.
6. Surveiller l'activité : surveillez l'activité de votre rôle de production à l'aide de CloudTrail journaux.

Ce qu'il faut savoir sur l'identité source

Gardez ce qui suit à l'esprit lorsque vous travaillez avec l'identité source.

- Les politiques d'approbation pour tous les rôles connectés à un fournisseur d'identité doivent disposer de l'autorisation `sts:SetSourceIdentity`. Pour les rôles qui ne disposent pas de cette autorisation dans la politique d'approbation de rôle, l'opération `AssumeRole*` échouera. Si vous ne souhaitez pas mettre à jour la politique d'approbation de rôle pour chaque rôle, vous pouvez utiliser une instance de fournisseur d'identité distincte pour transmettre l'identité source. Ajoutez ensuite l'autorisation `sts:SetSourceIdentity` uniquement aux rôles qui sont connectés au fournisseur d'identité distinct.
- Lorsqu'une identité définit une identité source, la clé `sts:SourceIdentity` est présente dans la demande. Pour les actions qui seront prises ultérieurement durant la session de rôle, la clé `aws:SourceIdentity` est présente dans la demande. AWS ne contrôle la valeur de l'identité source dans aucune des clés `sts:SourceIdentity` ou `aws:SourceIdentity`. Si vous choisissez d'exiger une identité source, vous devez choisir un attribut que vos utilisateurs ou votre IdP doivent fournir. Pour des raisons de sécurité, vous devez vérifier votre capacité à contrôler la façon dont ces valeurs sont fournies.
- La valeur de l'identité source doit comporter entre 2 et 64 caractères. Elle ne peut contenir que des caractères alphanumériques, des traits de soulignement et les caractères suivants : . , + = @ - (tiret). Vous ne pouvez pas utiliser une valeur qui commence par le texte **`aws:`**. Ce préfixe est réservé à un usage AWS interne.
- Les informations d'identité source ne sont pas capturées CloudTrail lorsqu'un AWS service ou un rôle lié à un service exécute une action pour le compte d'une identité fédérée ou d'un personnel.

> **Important**
>
> Vous ne pouvez pas passer à un rôle dans le AWS Management Console qui nécessite la définition d'une identité source lorsque le rôle est assumé. Pour assumer un tel rôle, vous pouvez utiliser l' AWS API AWS CLI or pour appeler l'`AssumeRole`opération et spécifier le paramètre d'identité source.

Autorisations requises pour définir l'identité source

En plus de l'action qui correspond à l'opération d'API, vous devez disposer de l'action d'autorisation suivante dans votre politique :

```
sts:SetSourceIdentity
```

- Pour spécifier une identité source, les principaux (utilisateurs et rôles IAM) doivent disposer des autorisations nécessaires pour `sts:SetSourceIdentity`. Votre qualité d'administrateur vous permet de configurer cela dans la politique de confiance de rôle et la politique d'autorisations du principal.
- Lorsque vous endossez un rôle avec un autre rôle (chaînage de rôles), des autorisations sont exigées pour `sts:SetSourceIdentity` tant dans la politique d'autorisations du principal qui endosse le rôle que dans la politique de confiance de rôle du rôle cible. Sinon, l'opération consistant à endosser le rôle échouera.
- Lorsque vous utilisez l'identité source, les politiques d'approbation de rôle pour tous les rôles connectés à un fournisseur d'identité doivent disposer de l'autorisation `sts:SetSourceIdentity`. L'opération `AssumeRole*` échouera pour tout rôle connecté à un fournisseur d'identité sans cette autorisation. Si vous ne voulez pas mettre à jour la politique de confiance de rôle pour chaque rôle, vous pouvez utiliser une instance IdP distincte pour transmettre l'identité source et ajouter l'autorisation `sts:SetSourceIdentity` aux seuls rôles qui sont connectés à l'instance IdP distincte.
- Pour définir une identité source au-delà des limites de compte, vous devez inclure l'autorisation `sts:SetSourceIdentity` à deux endroits. Elle doit être présente dans la politique d'autorisations du principal dans le compte d'origine et dans la politique de confiance de rôle du rôle dans le compte cible. Vous devrez peut-être faire cela lorsqu'un rôle est utilisé pour endosser un rôle dans un autre compte (chaînage de rôles).

Supposons, par exemple, qu'en tant qu'administrateur de compte, vous vouliez autoriser l'utilisateur IAM `DevUser` dans votre compte à endosser le `Developer_Role`dans le même compte. Mais vous voulez autoriser cette action uniquement si l'utilisateur a défini l'identité source à son nom d'utilisateur IAM. Vous pouvez attacher la politique suivante à l'utilisateur IAM.

Example Exemple de politique basée sur l'identité attachée à DevUser

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeRole",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
```

```
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role"
    },
    {
      "Sid": "SetAwsUserNameAsSourceIdentity",
      "Effect": "Allow",
      "Action": "sts:SetSourceIdentity",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role",
      "Condition": {
        "StringLike": {
          "sts:SourceIdentity": "${aws:username}"
        }
      }
    }
  ]
}
```

Pour appliquer les valeurs d'identité source acceptables, vous pouvez configurer la politique de confiance de rôle suivante. La politique donne à l'utilisateur IAM les autorisations `DevUser` pour endosser le rôle et définir une identité source. La clé de condition `sts:SourceIdentity` définit la valeur d'identité source acceptable.

Example Exemple de politique de confiance de rôle pour une identité source

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDevUserAssumeRole",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/DevUser"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "sts:SourceIdentity": "DevUser"
        }
      }
    }
  ]
```

```
}
```

À l'aide des informations d'identification de l'utilisateur IAM`DevUser`, celui-ci tente de supposer qu'il `DeveloperRole` utilise la AWS CLI demande suivante.

Example Exemple de demande AssumeRole CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Developer_Role \
--role-session-name Dev-project \
--source-identity DevUser \
```

Lors de l' AWS évaluation de la demande, le contexte de la demande contient le nom `sts:SourceIdentity` de`DevUser`.

Spécification d'une identité source lorsqu'un rôle est endossé

Vous pouvez spécifier une identité source lorsque vous utilisez l'une des opérations d' AWS STS `AssumeRole*`API pour obtenir des informations d'identification de sécurité temporaires pour un rôle. L'opération d'API que vous utilisez varie selon votre cas d'utilisation. Par exemple, si vous utilisez des rôles IAM pour donner aux utilisateurs IAM l'accès à AWS des ressources auxquelles ils n'ont normalement pas accès, vous pouvez utiliser cette opération. `AssumeRole` Si vous utilisez la fédération d'identités d'entreprise pour gérer vos utilisateurs de main-d'œuvre, vous pouvez utiliser l'opération `AssumeRoleWithSAML`. Si vous utilisez la fédération OIDC pour autoriser les utilisateurs finaux à accéder à vos applications mobiles ou Web, vous pouvez utiliser cette `AssumeRoleWithWebIdentity` opération. Les sections suivantes détaillent l'utilisation de l'identité source avec chaque opération. Pour en savoir plus sur les scénarios courants d'informations d'identification temporaires, veuillez consulter Scénarios courants d'informations d'identification temporaires.

Utilisation de l'identité source avec AssumeRole

L'`AssumeRole`opération renvoie un ensemble d'informations d'identification temporaires que vous pouvez utiliser pour accéder aux AWS ressources. Vous pouvez utiliser l'utilisateur ou les informations d'identification du rôle IAM pour appeler `AssumeRole`. Pour transmettre l'identité de la source tout en assumant un rôle, utilisez l'`--source-identity` AWS CLI option ou le paramètre `SourceIdentity` AWS API. L'exemple suivant vous explique comment spécifier l'identité source avec la AWS CLI.

Example Exemple de demande AssumeRole CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/developer \
--role-session-name Audit \
--source-identity Admin \
```

Utilisation de l'identité source avec AssumeRoleWith SAML

Le principal qui appelle l'opération `AssumeRoleWithSAML` est authentifié à l'aide de la fédération basée sur SAML. Cette opération renvoie un ensemble d'informations d'identification temporaires que vous pouvez utiliser pour accéder aux AWS ressources. Pour plus d'informations sur l'utilisation de la fédération basée sur SAML pour l' AWS Management Console accès, consultez. Permettre aux utilisateurs fédérés SAML 2.0 d'accéder au AWS Management Console Pour plus de détails sur AWS CLI AWS l'accès à l'API, consultezFédération SAML 2.0. Pour un didacticiel sur la configuration de la fédération SAML pour vos utilisateurs d'Active Directory, consultez la section Authentification AWS fédérée avec les services de fédération Active Directory (ADFS) dans le AWS blog de sécurité.

En tant qu'administrateur, vous pouvez autoriser les membres du répertoire de votre entreprise à se fédérer pour AWS utiliser cette AWS STS `AssumeRoleWithSAML` opération. Pour ce faire, effectuez les tâches suivantes :

1. Configurer un fournisseur SAML dans votre organisation.
2. Créer un fournisseur SAML dans IAM
3. Configurez un rôle et ses autorisations AWS pour vos utilisateurs fédérés.
4. Achever la configuration de l'IdP SAML et créer des assertions pour la réponse d'authentification SAML.

Pour définir un attribut SAML pour l'identité source, incluez dans l'élément `Attribute` l'attribut `Name` défini à l'adresse `https://aws.amazon.com/SAML/Attributes/SourceIdentity`. Utilisez l'élément `AttributeValue` pour spécifier la valeur de l'identité source. Par exemple, supposons que vous souhaitez transmettre l'attribut d'identité suivant en tant qu'identité source :

`SourceIdentity:DiegoRamirez`

Pour transmettre cet attribut, incluez l'élément suivant dans votre assertion SAML.

Example Extrait d'une assertion SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
<AttributeValue>DiegoRamirez</AttributeValue>
</Attribute>
```

Utilisation de l'identité source avec AssumeRoleWithWebIdentity

Le principal appelant l'`AssumeRoleWithWebIdentity`opération est authentifié à l'aide d'une fédération conforme à OpenID Connect (OIDC). Cette opération renvoie un ensemble d'informations d'identification temporaires que vous pouvez utiliser pour accéder aux ressources AWS . Pour plus d'informations sur l'utilisation de la fédération OIDC pour AWS Management Console l'accès, consultezFédération OIDC.

Pour transmettre l'identité source depuis OpenID Connect (OIDC), vous devez inclure l'identité source dans le jeton web JSON (JWT). Incluez l'identité source dans l'espace de noms `https://aws.amazon.com/` `source_identity` dans le jeton lorsque vous soumettez la demande `AssumeRoleWithWebIdentity`. Pour en savoir plus sur les jetons OIDC et les revendications, veuillez consulter Utilisation des jetons avec les groupes d'utilisateurs dans le Guide du développeur Amazon Cognito .

Par exemple, le JWT décodé suivant est un jeton utilisé pour appeler `AssumeRoleWithWebIdentity` avec l'identité source `Admin`.

Example Exemple de jeton web JSON décodé

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/source_identity":"Admin"
}
```

Contrôle de l'accès au moyen des informations d'identité source

Lorsqu'une identité source est initialement définie, la SourceIdentity clé sts : est présente dans la requête. Une fois qu'une identité source est définie, la SourceIdentity clé aws : est présente dans

toutes les demandes suivantes effectuées au cours de la session de rôle. En tant qu'administrateur, vous pouvez rédiger des politiques qui accordent une autorisation conditionnelle pour effectuer des AWS actions en fonction de l'existence ou de la valeur de l'attribut d'identité source.

Imaginez que vous souhaitiez demander à vos développeurs de définir une identité source afin d'assumer un rôle essentiel et d'autoriser l'écriture sur une AWS ressource critique de production. Imaginez également que vous accordez AWS l'accès à l'identité de votre personnel en utilisant`AssumeRoleWithSAML`. Comme vous voulez que seuls les développeurs senior Saanvi et Diego aient accès au rôle, vous créez a politique de confiance suivante pour le rôle.

Example Exemple de politique d'approbation de rôle pour une identité source (SAML)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "SAMLProviderAssumeRoleWithSAML",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:AssumeRoleWithSAML"
      ],
      "Condition": {
        "StringEquals": {
          "SAML:aud": "https://signin.aws.amazon.com/saml"
        }
      }
    },
    {
      "Sid": "SetSourceIdentitySrEngs",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:SetSourceIdentity"
      ],
      "Condition": {
```

```
        "StringLike": {
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
        }
      }
    }
  ]
}
```

La politique d'approbation contient une condition pour l'interface `sts:SourceIdentity` qui exige une identité source de Saanvi ou Diego pour endosser le rôle critique.

Sinon, si vous utilisez un fournisseur OIDC pour la fédération et que les utilisateurs sont authentifiés`AssumeRoleWithWebIdentity`, votre politique de confiance dans les rôles peut se présenter comme suit.

Example Exemple de politique d'approbation de rôle pour l'identité source (fournisseur OIDC)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/server.example.com"
      },
      "Action": [
        "sts:AssumeRoleWithWebIdentity",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "server.example.com:aud": "oidc-audience-id"
        },
        "StringLike": {
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
        }
      }
```

```
    }
  ]
}
```

Chaînage de rôles et exigences entre comptes

Supposons que vous vouliez autoriser les utilisateurs ayant endossé un `CriticalRole` à endosser un `CriticalRole_2` dans un autre compte. Les informations d'identification de session de rôle qui ont été obtenues pour endosser le `CriticalRole` sont utilisées pour effectuer un chaînage de rôles à un second rôle, `CriticalRole_2`, dans un autre compte. Le rôle est endossé au-delà d'une limite de compte. Par conséquent, l'autorisation `sts:SetSourceIdentity` doit être octroyée tant dans la politique d'autorisations sur le `CriticalRole` que dans la politique de confiance de rôle sur le `CriticalRole_2`.

Example Exemple de politique d'autorisation sur CriticalRole

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeRoleAndSetSourceIdentity",
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Resource": "arn:aws:iam::222222222222:role/CriticalRole_2"
    }
  ]
}
```

Afin de sécuriser la définition de l'identité source au-delà de la limite du compte, la politique de confiance de rôle suivante fait uniquement confiance au principal de rôle pour `CriticalRole` pour définir l'identité source.

Example Exemple de politique de confiance dans les rôles sur CriticalRole _2

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111111111111:role/CriticalRole"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
          "aws:SourceIdentity": ["Saanvi","Diego"]
        }
      }
    }
  ]
}
```

L'utilisateur effectue l'appel suivant en utilisant les informations d'identification de session de rôle obtenues en assumant CriticalRole. L'identité de la source a été définie lors de l'hypothèse de CriticalRole, il n'est donc pas nécessaire de la redéfinir explicitement. Si l'utilisateur tente de définir une identité source différente de la valeur définie lors de la supposition de `CriticalRole`, la demande d'endosser le rôle sera refusée.

Example Exemple de demande AssumeRole CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::222222222222:role/CriticalRole_2 \
--role-session-name Audit \
```

Lorsque le principal appelant endosse le rôle, l'identité source dans la demande persiste à partir de la première session de rôle endossée. Par conséquent, les clés `aws:SourceIdentity` et `sts:SourceIdentity` sont toutes deux présentes dans le contexte de demande.

Affichage de l'identité de la source dans CloudTrail

Vous pouvez l'utiliser CloudTrail pour afficher les demandes faites pour assumer des rôles ou fédérer des utilisateurs. Vous pouvez également afficher les demandes de rôle ou d'utilisateur souhaitant effectuer des actions dans AWS. Le fichier CloudTrail journal contient des informations sur l'identité source définie pour le rôle assumé ou la session utilisateur fédérée. Pour plus d'informations, consultez Journalisation des appels IAM et AWS STS API avec AWS CloudTrail.

Supposons, par exemple, qu'un utilisateur fasse une AWS STS `AssumeRole` demande et définisse une identité source. Vous pouvez trouver les `sourceIdentity` informations dans la `requestParameters` clé de votre CloudTrail journal.

Example Exemple de section RequestParameters dans un journal AWS CloudTrail

```
"eventVersion": "1.05",
    "userIdentity": {
        "type": "AWSAccount",
        "principalId": "AIDAJ45Q7YFFAREXAMPLE",
        "accountId": "111122223333"
    },
    "eventTime": "2020-04-02T18:20:53Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.64",
    "userAgent": "aws-cli/1.16.96 Python/3.6.0 Windows/10 botocore/1.12.86",
    "requestParameters": {
        "roleArn": "arn:aws:iam::123456789012:role/DevRole",
        "roleSessionName": "Dev1",
        "sourceIdentity": "source-identity-value-set"
    }
```

Si l'utilisateur utilise la session de rôle assumé pour effectuer une action, les informations d'identité de la source sont présentes dans la `userIdentity` clé du CloudTrail journal.

Example Exemple de clé UserIdentity dans un journal AWS CloudTrail

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAJ45Q7YFFAREXAMPLE:Dev1",
    "arn": "arn:aws:sts::123456789012:assumed-role/DevRole/Dev1",
    "accountId": "123456789012",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAJ45Q7YFFAREXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/DevRole",
```

```
        "accountId": "123456789012",
        "userName": "DevRole"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-02-21T23:46:28Z"
      },
      "sourceIdentity": "source-identity-value-present"
    }
  }
}
```

Pour voir des exemples AWS STS d'événements d'API dans CloudTrail les journaux, consultezExemple d'événements d'API IAM dans le journal CloudTrail. Pour plus de détails sur les informations contenues dans les fichiers CloudTrail journaux, consultez la section Référence des CloudTrail événements dans le guide de AWS CloudTrail l'utilisateur.

Autorisations pour GetFederationToken

L'opération `GetFederationToken` est appelée par un utilisateur IAM et renvoie les informations d'identification temporaires pour cet utilisateur. Cette opération fédère l'utilisateur. Les autorisations accordées à un utilisateur fédéré sont définies dans l'un des éléments suivants :

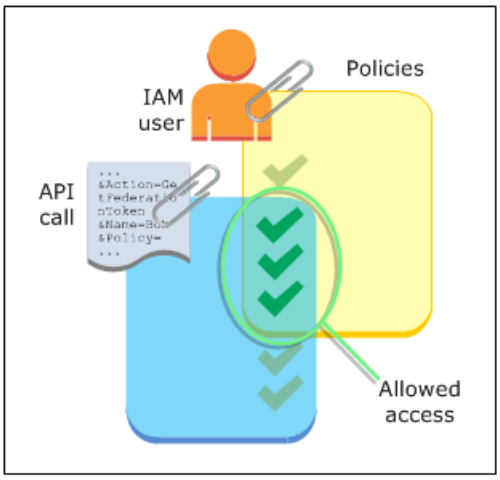
- Les politiques de session transmises en tant que paramètre de l'appel d'API `GetFederationToken`. (Il s'agit du scénario le plus courant.)
- Une politique basée sur les ressources qui nomme explicitement l'utilisateur fédéré dans l'élément `Principal` de la politique. (Ce scénario est moins courant.)

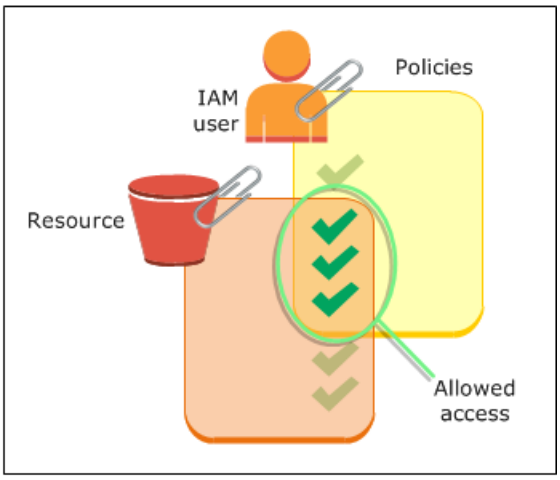
Les politiques de session sont des politiques avancées que vous transmettez en tant que paramètres lorsque vous créez par programmation une session temporaire. Lorsque vous créez une session d'utilisateur fédéré et transmettez des stratégies de session, les autorisations de la session obtenues sont une combinaison de la stratégie basée sur l'identité de l'utilisateur et des stratégies de session. Vous ne pouvez pas utiliser la politique de session pour accorder plus d'autorisations que celles autorisées par la politique basée sur l'identité de l'utilisateur fédéré.

Cela signifie que dans la plupart des cas, si vous ne transmettez pas de politique avec l'appel d'API `GetFederationToken`, les informations d'identification de sécurité temporaires générées ne disposent d'aucune autorisation. Toutefois, une politique basée sur les ressources peut fournir des

autorisations supplémentaires pour la session. Vous pouvez accéder à une ressource avec une politique basée sur les ressources, qui spécifie votre session en tant que principal autorisé.

Les illustrations suivantes sont une représentation visuelle de la façon dont les politiques interagissent pour déterminer les autorisations accordées aux informations d'identification de sécurité temporaires retournées par un appel à `GetFederationToken`.





Exemple : attribution d'autorisations à l'aide de GetFederationToken

Vous pouvez utiliser l'action d'API `GetFederationToken` avec différents types de politiques. Voici quelques exemples.

Politique attachée à l'utilisateur IAM

Dans cet exemple, une application cliente basée sur un navigateur s'appuie sur deux services web backend. L'un des backend est votre propre serveur d'authentification qui utilise votre système d'identité pour authentifier l'application cliente. L'autre backend est un service AWS qui fournit certaines des fonctionnalités de l'application cliente. L'application cliente est authentifiée par votre serveur, puis ce dernier créée ou récupère la politique d'autorisation appropriée. Ensuite, votre serveur appelle l'API `GetFederationToken` afin d'obtenir des informations d'identification de sécurité temporaires, puis il retourne ces informations d'identification à l'application cliente. L'application cliente peut ensuite adresser des demandes directement au AWS service avec les informations d'identification de sécurité temporaires. Cette architecture permet à l'application cliente de faire des AWS demandes sans intégrer d'informations d' AWS identification à long terme.

Votre serveur d'authentification appelle l'API `GetFederationToken` avec les informations d'identification de sécurité à long terme d'un utilisateur IAM nommé `token-app`. Cependant, les informations d'identification de l'utilisateur IAM à long terme restent sur votre serveur et ne sont

jamais distribuées au client. L'exemple de politique suivant est attaché à l'utilisateur IAM `token-app` et définit le plus large ensemble d'autorisations dont vos utilisateurs fédérés (clients) auront besoin. Notez que l'autorisation `sts:GetFederationToken` est requise pour que votre service d'authentification puisse obtenir les informations d'identification de sécurité temporaires pour les utilisateurs fédérés.

> **Note**
>
> AWS fournit un exemple d'application Java à cette fin, que vous pouvez télécharger ici : Token Vending Machine for Identity Registration - Sample Java Web Application.

Example Exemple de politique attachée à un **`token-app`** d'utilisateur IAM qui appelle **`GetFederationToken`**

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:GetFederationToken",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "dynamodb:ListTables",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sqs:ReceiveMessage",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sns:ListSubscriptions",
      "Resource": "*"
```

```
    }
  ]
}
```

La politique précédente accorde plusieurs autorisations à l'utilisateur IAM. Cependant, cette politique seule n'accorde pas d'autorisations à l'utilisateur fédéré. Si cet utilisateur IAM appelle `GetFederationToken` et ne transmet pas de politique en tant que paramètre de l'appel d'API, l'utilisateur fédéré qui en résulte n'a aucune autorisation effective.

Politique de session transmise en tant que paramètre

La méthode la plus courante pour s'assurer que l'autorisation appropriée est octroyée à l'utilisateur fédéré consiste à transmettre des politiques de session de l'appel d'API `GetFederationToken`. En reprenant l'exemple précédent, imaginons que `GetFederationToken` est appelé avec les informations d'identification de l'utilisateur `token-app` IAM. Imaginons ensuite que la politique de session suivante est transmise en tant que paramètre de l'appel d'API. L'utilisateur fédéré a obtenu l'autorisation de répertorier le contenu du compartiment Amazon S3 nommé `productionapp`. L'utilisateur ne peut pas effectuer les actions Amazon S3 `GetObject` `PutObject` et `DeleteObject` sur les éléments dans le compartiment `productionapp`.

L'utilisateur fédéré se voit attribuer ces autorisations, car elles sont une combinaison des politiques d'utilisateur IAM et les politiques de session que vous transmettez.

L'utilisateur fédéré n'a pas pu effectuer d'actions dans Amazon SNS, Amazon SQS, Amazon DynamoDB, ni dans un compartiment S3 à l'exception de `productionapp`. Ces actions sont refusées, même si ces autorisations sont accordées à l'utilisateur IAM associé à l'appel `GetFederationToken`.

Example Exemple de politique transmise en tant que paramètre de l'appel d'API **`GetFederationToken`**

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::productionapp"]
    },
    {
```

```
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": ["arn:aws:s3:::productionapp/*"]
    }
  ]
}
```

Politiques basées sur les ressources

Certaines AWS ressources prennent en charge les politiques basées sur les ressources, et ces politiques fournissent un autre mécanisme permettant d'accorder des autorisations directement à un utilisateur fédéré. Seuls certains AWS services prennent en charge les politiques basées sur les ressources. Par exemple, Amazon S3 comprend des compartiments, Amazon SNS des rubriques, et Amazon SQS des files d'attente auxquelles vous pouvez attacher des politiques. Pour obtenir la liste de tous les services prenant en charge les politiques basées sur les ressources, consultez AWS services qui fonctionnent avec IAM et passez en revue la colonne « Politiques basées sur les ressources » des tableaux. Vous pouvez utiliser des politiques basées sur les ressources pour attribuer des autorisations directement à un utilisateur fédéré. Pour ce faire, vous devez spécifier l'Amazon Resource Name (ARN) de l'utilisateur fédéré dans l'élément `Principal` de la politique basée sur les ressources. L'exemple suivant illustre cette méthode et développe les exemples précédents, à l'aide d'un compartiment S3 nommé `productionapp`.

La politique basée sur les ressources suivante est attachée au compartiment. Cette politique de compartiment permet à un utilisateur fédéré nommé Carol d'accéder au compartiment. Lorsque la politique d'exemple décrite précédemment est attachée à l'utilisateur IAM `token-app`, l'utilisateur fédéré nommé Carol est autorisé à effectuer les actions `s3:GetObject`, `s3:PutObject` et `s3:DeleteObject` sur le compartiment nommé `productionapp`. Cela est vrai même lorsqu'aucune politique de session n'est transmise en tant que paramètre de l'appel d'API `GetFederationToken`. En effet, dans ce cas, la politique suivante basée sur les ressources a octroyé explicitement des autorisations à l'utilisateur fédéré nommé Carol.

Gardez à l'esprit que les autorisations ne sont accordées à un utilisateur fédéré que si celles-ci sont explicitement accordées à la fois à l'utilisateur IAM et à l'utilisateur fédéré. Ils peuvent également être accordés (au sein du compte) par une politique basée sur les ressources qui nomme explicitement l'utilisateur fédéré dans l'élément `Principal` de la politique, comme dans l'exemple suivant.

Example Exemple de politique de compartiment qui autorise l'accès à l'utilisateur fédéré

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Principal": {"AWS": "arn:aws:sts::account-id:federated-user/Carol"},
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": ["arn:aws:s3:::productionapp/*"]
  }
}
```

Pour plus d'informations sur la manière dont les politiques sont évaluées, consultez la rubrique Logique d'évaluation des politiques.

Autorisations pour GetSessionToken

La bonne occasion pour appeler l'`GetSessionToken` opération d'API ou la `get-session-token` commande CLI est lorsqu'un utilisateur doit être authentifié avec l'authentification multi-facteur (MFA). Il est possible d'écrire une politique qui autorise certaines actions uniquement lorsque ces actions sont requises par un utilisateur authentifié à l'aide de MFA. Pour réussir à transmettre le contrôle de l'autorisation d'authentification MFA, un utilisateur doit d'abord appeler `GetSessionToken` et inclure le `SerialNumber` facultatif et `TokenCode` les paramètres. Si l'utilisateur est correctement authentifié par un dispositif MFA, les informations d'identification retournées par l'opération d'API `GetSessionToken` incluent le contexte MFA. Ce contexte indique que l'utilisateur est authentifié par l'authentification MFA et est autorisé pour les opérations d'API nécessitant une authentification MFA.

Autorisations requises pour GetSessionToken

Aucune autorisation n'est requise pour qu'un utilisateur obtienne un jeton de session. Le but principal de l'opération `GetSessionToken` est d'authentifier l'utilisateur à l'aide de l'authentification MFA. Vous ne pouvez pas utiliser de stratégies pour contrôler les opérations d'authentification.

Pour autoriser l'exécution de la plupart AWS des opérations, vous devez ajouter l'action portant le même nom à une politique. Par exemple, pour créer un utilisateur, vous devez utiliser l'opération d'API `CreateUser`, la commande CLI `create-user` ou l' AWS Management Console. Pour

effectuer ces opérations, vous devez disposer d'une politique qui vous permet d'accéder à l'action `CreateUser` .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateUser",
            "Resource": "*"
        }
    ]
}
```

Vous pouvez inclure l'action `GetSessionToken` dans vos politiques, mais elle n'a aucun effet sur la possibilité de l'utilisateur d'effectuer l'opération `GetSessionToken` .

Autorisations octroyées par GetSessionToken

Si `GetSessionToken` est appelée avec les informations d'identification d'un utilisateur IAM, les informations d'identification de sécurité temporaires ont les mêmes autorisations que cet utilisateur IAM. De même, `GetSessionToken` s'ils sont appelés avec des Utilisateur racine d'un compte AWS informations d'identification, les informations d'identification de sécurité temporaires ont des autorisations d'utilisateur root.

> **Note**
>
> Il est recommandé de ne pas appeler `GetSessionToken` avec les informations d'identification d'utilisateur racine. Mettez plutôt en application nos bonnes pratiques pour créer des utilisateurs IAM avec les autorisations nécessaires. Utilisez ensuite ces utilisateurs IAM pour vos interactions quotidiennes avec AWS.

Les informations d'identification temporaires que vous obtenez lorsque vous appelez `GetSessionToken` présentent les fonctionnalités et limitations suivantes :

- Vous pouvez utiliser les informations d'identification pour accéder au AWS Management Console en les transmettant au point de terminaison d'authentification unique de la fédération à l'`https://signin.aws.amazon.com/federation`adresse. Pour plus d'informations, consultez Activation de l'accès à la AWS console par un courtier d'identité personnalisé.

- Vous ne pouvez pas utiliser les informations d'identification pour appeler les opérations IAM ou d'API AWS STS . Vous pouvez les utiliser pour appeler des opérations d'API pour d'autres AWS services.

Comparez cette opération d'API et ses limitations et capacités avec les autres opérations d'API qui créent des informations d'identification de sécurité temporaires à Comparaison des opérations AWS STS d'API

Pour plus d'informations sur l'accès aux API protégé par MFA à l'aide de `GetSessionToken`, consultez Configuration de l'accès aux API protégé par MFA.

Désactivation des autorisations affectées aux informations d'identification de sécurité temporaires

Les informations d'identification de sécurité temporaires sont valides jusqu'à la fin de leur délai d'expiration. Ces informations d'identification sont valides pendant la durée spécifiée, de 900 secondes (15 minutes) à un maximum de 129 600 secondes (36 heures). La durée de session par défaut est de 43 200 secondes (12 heures). Vous pouvez révoquer ces informations d'identification, mais vous devez également modifier les autorisations associées au rôle afin de mettre fin à l'utilisation d'informations d'identification compromises pour des activités malveillantes sur le compte. Les autorisations attribuées aux informations d'identification de sécurité temporaires sont évaluées chaque fois qu'elles sont utilisées pour faire une AWS demande. Une fois que vous avez supprimé toutes les autorisations des informations d'identification, les AWS demandes qui les utilisent échouent.

Les mises à jour des politiques peuvent prendre quelques minutes pour être mises en vigueur. Révoquer les informations d'identification de sécurité temporaires du rôle pour obliger tous les utilisateurs endossant ce rôle à s'authentifier à nouveau et à demander de nouvelles informations d'identification.

Vous ne pouvez pas modifier les autorisations pour un Utilisateur racine d'un compte AWS. De même, vous ne pouvez pas modifier les autorisations des informations d'identification de sécurité temporaires créées en appelant `GetFederationToken` ou `GetSessionToken` lorsque vous êtes connecté comme utilisateur racine. C'est pour cette raison que nous vous recommandons de ne pas appeler `GetFederationToken` ou `GetSessionToken` en tant qu'utilisateur racine.

> **Important**
>
> Vous ne pouvez pas modifier les rôles dans IAM qui ont été créés à partir des ensembles d'autorisations IAM Identity Center. Vous devez révoquer la session d'ensemble d'autorisations active pour un utilisateur dans IAM Identity Center. Pour plus d'informations, voir Révoquer les sessions de rôle IAM actives créées par des ensembles d'autorisations dans le guide de l'utilisateur d'IAM Identity Center.

Rubriques

- Refuser l'accès à toutes les sessions associées à un rôle
- Refuser l'accès à une session spécifique
- Refuser une session d'utilisateur à l'aide de clés de contexte de condition
- Refuser un utilisateur de session avec des politiques basées sur les ressources

Refuser l'accès à toutes les sessions associées à un rôle

Utilisez cette méthode lorsque vous craignez un accès suspect des :

- Principaux d'un autre compte utilisant l'accès intercompte
- Identités des utilisateurs externes autorisés à accéder aux AWS ressources de votre compte
- Utilisateurs authentifiés dans une application mobile ou Web auprès d'un fournisseur OIDC

Cette procédure refuse les autorisations pour tous utilisateurs autorisés à endosser un rôle.

Pour modifier ou supprimer les autorisations affectées aux informations d'identification de sécurité temporaires obtenues en appelant `AssumeRole`, `AssumeRoleWithSAML`, `AssumeRoleWithWebIdentity`, `GetFederationToken` ou `GetSessionToken`, vous pouvez modifier ou supprimer la politique d'autorisations qui définit les autorisations du rôle.

> **Important**
>
> S'il existe une politique basée sur les ressources qui autorise l'accès au principal, vous devez également ajouter un refus explicite pour cette ressource. Consultez Refuser un utilisateur de session avec des politiques basées sur les ressources pour plus de détails.

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la.
2. Dans le panneau de navigation, choisissez le nom du rôle à modifier. Vous pouvez utiliser la zone de recherche pour filtrer la liste.
3. Sélectionnez la politique concernée.
4. Choisissez l'onglet Permissions (Autorisations).
5. Choisissez l'onglet JSON et mettez à jour la politique pour refuser toutes les ressources et actions.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*"
    }
  ]
}
```

6. Sur la page Vérification, vérifiez le Récapitulatif de la politique, puis choisissez Enregistrer les modifications pour enregistrer votre travail.

Lorsque vous mettez à jour la politique, les modifications affectent les autorisations de toutes les informations d'identification de sécurité temporaires associées au rôle, y compris les informations d'identification émises avant que vous apportiez des changements à la politique d'autorisations du rôle. Après avoir mis à jour la politique, vous pouvez révoquer les informations d'identification de sécurité temporaires du rôle pour révoquer immédiatement toutes les autorisations relatives aux informations d'identification émises pour le rôle.

Refuser l'accès à une session spécifique

Lorsque vous mettez à jour les rôles qui peuvent être endossés par un IdP à l'aide d'une politique de refus ou que vous supprimez complètement le rôle, tous les utilisateurs ayant accès au rôle sont perturbés. Vous pouvez refuser l'accès en fonction de l'élément `Principal` sans affecter les autorisations de toutes les autres sessions associées au rôle.

Le `Principal` peut se voir refuser des autorisations à l'aide des clés de contexte de condition ou des politiques basées sur les ressources.

> **Tip**
>
> Vous pouvez trouver les ARN des utilisateurs fédérés à l'aide AWS CloudTrail des journaux. Pour plus d'informations, consultez Comment identifier facilement vos utilisateurs fédérés à l'aide AWS CloudTrail de.

Refuser une session d'utilisateur à l'aide de clés de contexte de condition

Vous pouvez utiliser des clés de contexte de condition lorsque vous souhaitez refuser l'accès aux informations d'identification de sécurité temporaires sans affecter les autorisations de l'utilisateur ou du rôle IAM ayant créé les informations d'identification.

Pour plus d'informations sur les clés de contexte de condition, veuillez consulter la rubrique AWS clés contextuelles de condition globale.

> **Note**
>
> S'il existe une politique basée sur les ressources qui autorise l'accès au principal, vous devez également ajouter une instruction de refus explicite sur la politique basée sur les ressources après ces étapes.

Après avoir mis à jour la politique, vous pouvez révoquer les informations d'identification de sécurité temporaires du rôle pour révoquer immédiatement toutes les informations d'identification émises.

lois : PrincipalArn

Vous pouvez utiliser la clé de contexte de condition lois : PrincipalArn pour refuser l'accès à un ARN principal spécifique. Pour ce faire, vous devez spécifier l'identifiant unique (ID) de l'utilisateur, du rôle ou de l'utilisateur fédéré IAM auxquels les informations d'identification de sécurité temporaires sont associées dans l'élément Condition d'une politique.

1. Dans le panneau de navigation de la console IAM, choisissez le nom du rôle à modifier. Vous pouvez utiliser la zone de recherche pour filtrer la liste.
2. Sélectionnez la politique concernée.
3. Choisissez l’onglet Permissions (Autorisations).
4. Choisissez l'onglet JSON et ajoutez une instruction de refus pour l'ARN principal, comme indiqué dans l'exemple suivant.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:role/ROLENAME",
            "arn:aws:iam::222222222222:user/USERNAME",
            "arn:aws:sts::222222222222:federated-user/USERNAME"
          ]
        }
      }
    }
  ]
}
```

5. Sur la page Vérification, vérifiez le Récapitulatif de la politique, puis choisissez Enregistrer les modifications pour enregistrer votre travail.

aws:userid

Vous pouvez utiliser la clé de contexte de condition aws:userid pour refuser l'accès à toutes les sessions d'informations d'identification de sécurité temporaires ou à certaines d'entre elles associées à l'utilisateur ou au rôle IAM. Pour ce faire, vous devez spécifier l'identifiant unique (ID) de l'utilisateur, du rôle ou de l'utilisateur fédéré IAM auxquels les informations d'identification de sécurité temporaires sont associées dans l'élément `Condition` d'une politique.

La politique suivante montre un exemple de la manière dont vous pouvez refuser l'accès à des sessions d'informations d'identification de sécurité temporaires à l'aide d'une clé de contexte de condition `aws:userid`.

- `AIDAXUSER1` représente l'identifiant unique d'un utilisateur IAM. En spécifiant l'identifiant unique d'un utilisateur IAM en tant que valeur de la clé de contexte `aws:userid`, vous refuserez toutes les sessions associées à l'utilisateur IAM.

- `AROAXROLE1` représente l'identifiant unique d'un rôle IAM. En spécifiant l'identifiant unique d'un rôle IAM en tant que valeur de la clé de contexte `aws:userid`, vous refuserez toutes les sessions associées au rôle.
- `AROAXROLE2` représente l'identifiant unique d'une session de rôle endossé. Dans la partie caller-specified-role-session -name de l'identifiant unique du rôle assumé, vous pouvez spécifier un rôle, un nom de session ou un caractère générique si l' StringLikeopérateur de condition est utilisé. Si vous spécifiez le nom de la session de rôle, cela refusera la session de rôle nommée sans affecter les autorisations du rôle qui a créé les informations d'identification. Si vous spécifiez un caractère générique pour le nom de session du rôle, toutes les sessions associées au rôle seront refusées.
- `account-id:<federated-user-caller-specified-name>` représente l'identifiant unique d'une session d'utilisateur fédéré. Un utilisateur fédéré est créé par un utilisateur IAM qui appelle l' GetFederationToken API. Si vous spécifiez l'identifiant unique d'un utilisateur fédéré, cela refusera la session d'utilisateur fédéré nommée sans affecter les autorisations du rôle qui a créé les informations d'identification.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:userId": [
            "AIDAXUSER1",
            "AROAXROLE1",
            "AROAXROLE2:<caller-specified-role-session-name>",
            "account-id:<federated-user-caller-specified-name>"
          ]
        }
      }
    }
  ]
}
```

Pour des exemples spécifiques de valeurs de clé de principal, veuillez consulter la rubrique Valeurs de la clé du principal. Pour plus d'informations sur les identifiants uniques IAM, veuillez consulter la rubrique Identifiants uniques.

Refuser un utilisateur de session avec des politiques basées sur les ressources

Si l'ARN principal est également inclus dans des politiques basées sur les ressources, vous devez également révoquer l'accès en fonction des valeurs `principalId` ou `sourceIdentity` de l'utilisateur spécifique dans l'élément `Principal` d'une politique basée sur les ressources. Si vous mettez uniquement à jour la politique d'autorisations pour le rôle, l'utilisateur peut toujours effectuer les actions autorisées dans la politique basée sur les ressources.

1. Reportez-vous à la rubrique AWS services qui fonctionnent avec IAM pour voir si le service prend en charge les politiques basées sur les ressources.
2. Connectez-vous à la console du service AWS Management Console et ouvrez-la. Chaque service dispose d'un emplacement différent dans la console pour associer des politiques.
3. Modifiez l'instruction de politique pour spécifier les informations d'identification de l'identifiant :
   a. Dans `Principal`, saisissez l'ARN de l'information d'identification à refuser.
   b. Dans `Effect`, saisissez « Deny. » (Refuser).
   c. Dans `Action`, saisissez l'espace de noms du service et le nom de l'action à refuser. Pour refuser toutes les actions, utilisez le caractère générique (*). Par exemple : « s3:*. ».
   d. Dans `Resource`, saisissez l'ARN de la ressource cible. Par exemple : « arn:aws:s3:::EXAMPLE-BUCKET. ».

```
{
"Version": "2012-10-17",
  "Statement": {
    "Principal": [
            "arn:aws:iam::222222222222:role/ROLENAME",
            "arn:aws:iam::222222222222:user/USERNAME",
            "arn:aws:sts::222222222222:federated-user/USERNAME"
    ],
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::EXAMPLE-BUCKET"
  }
}
```

4. Enregistrez votre travail.

Octroi d'autorisations pour créer des informations d'identification de sécurité temporaires

Par défaut, les utilisateurs IAM n'ont pas l'autorisation de créer des informations d'identification de sécurité temporaires pour des utilisateurs fédérés et les rôles. Vous devez utiliser une politique pour fournir ces autorisations à vos utilisateurs. Bien que vous puissiez accorder des autorisations directement à un utilisateur, nous vous recommandons vivement d'accorder des autorisations à un groupe. Cela facilite la gestion des autorisations. Lorsqu'un utilisateur n'a plus besoin d'effectuer les tâches associées aux autorisations, il vous suffit de le supprimer du groupe. Si un autre utilisateur doit effectuer cette tâche, ajoutez-le au groupe pour lui accorder les autorisations.

Pour accorder à un groupe IAM l'autorisation de créer des informations d'identification de sécurité temporaires pour des utilisateurs fédérés ou des groupes, vous attachez une politique qui accorde un ou plusieurs des privilèges suivants :

- Pour que des utilisateurs fédérés accèdent à un rôle IAM, accordez l'accès à AWS STS `AssumeRole`.
- Pour les utilisateurs fédérés qui n'ont pas besoin de rôle, accordez l'accès à AWS STS `GetFederationToken`.

Pour plus d'informations sur les différences entre les opérations d'API `AssumeRole` et `GetFederationToken`, consultez Demande d'informations d'identification temporaires de sécurité.

Les utilisateurs IAM peuvent également appeler `GetSessionToken` pour créer des informations d'identification de sécurité temporaires. Aucune autorisation n'est requise pour qu'un utilisateur puisse appeler `GetSessionToken`. Le but principal de cette opération est d'authentifier l'utilisateur à l'aide de l'authentification MFA. Vous ne pouvez pas utiliser les politiques pour contrôler l'authentification. Cela signifie que vous ne pouvez pas empêcher les utilisateurs IAM d'appeler `GetSessionToken` pour créer des informations d'identification temporaires.

Example Exemple de politique qui accorde l'autorisation d'endosser un rôle

L'exemple de politique suivant accorde l'autorisation d'appeler `AssumeRole` pour le `UpdateApp` rôle dans Compte AWS `123123123123`. Quand `AssumeRole` est utilisé, l'utilisateur (ou l'application) qui crée les informations d'identification de sécurité pour le compte d'un utilisateur fédéré ne peut déléguer aucune autorisation non spécifiée dans la politique d'autorisation de rôle.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::123123123123:role/UpdateAPP"
  }]
}
```

Example Exemple de politique qui accorde l'autorisation de créer des informations d'identification de sécurité temporaires pour un utilisateur fédéré

L'exemple de politique suivant donne l'autorisations d'accéder à `GetFederationToken`.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": "*"
  }]
}
```

> **Important**
>
> Lorsque vous accordez à un utilisateur IAM l'autorisation de créer des informations d'identification de sécurité temporaires pour des utilisateurs fédérés avec `GetFederationToken`, sachez que cela lui autorise à déléguer ses propres autorisations. Pour plus d'informations sur la délégation d'autorisations entre les utilisateurs IAM et consultez Comptes AWS. Exemples de politiques pour la délégation d'accès Pour plus d'informations sur le contrôle des autorisations dans les informations d'identification de sécurité temporaires, consultez Contrôle des autorisations affectées aux informations d'identification de sécurité temporaires.

Example Exemple de politique qui accorde à un utilisateur une autorisation limitée de créer des informations d'identification de sécurité temporaires pour des utilisateurs fédérés

Lorsque vous laissez un utilisateur IAM appeler `GetFederationToken`, il s'agit d'une bonne pratique pour limiter les autorisations que l'utilisateur IAM peut déléguer. Par exemple, la politique suivante explique comment autoriser un utilisateur IAM à créer des informations d'identification

de sécurité temporaires uniquement pour des utilisateurs fédérés dont les noms commencent par Manager.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": ["arn:aws:sts::123456789012:federated-user/Manager*"]
  }]
}
```

Octroi d'autorisations pour utiliser des sessions de console sensibles à l'identité

Les sessions de console basées sur l'identité permettent aux AWS IAM Identity Center utilisateurs et aux identifiants de session d'être inclus dans les sessions de AWS console des utilisateurs lorsqu'ils se connectent. Par exemple, Amazon Q Developer Pro utilise des sessions de console sensibles à l'identité pour personnaliser l'expérience de service. Pour plus d'informations sur les sessions de console sensibles à l'identité, consultez la section Activation des sessions de console basées sur l'identité dans le Guide de l'utilisateur.AWS IAM Identity Center Pour plus d'informations sur la configuration d'Amazon Q Developer, consultez la section Configuration d'Amazon Q Developer dans le manuel Amazon Q Developer User Guide.

Pour que les sessions de console sensibles à l'identité soient accessibles à un utilisateur, vous devez utiliser une politique basée sur l'identité pour accorder au principal IAM l'`sts:SetContext`autorisation d'accéder à la ressource qui représente sa propre session de console.

> **Important**
>
> Par défaut, les utilisateurs ne sont pas autorisés à définir le contexte de leurs sessions de console basées sur l'identité. Pour ce faire, vous devez accorder au principal IAM l'`sts:SetContext`autorisation dans le cadre d'une politique basée sur l'identité, comme indiqué dans l'exemple de stratégie ci-dessous.

L'exemple de politique basée sur l'identité suivant accorde l'`sts:SetContext`autorisation à un principal IAM, ce qui permet au principal de définir un contexte de session de console sensible à l'identité pour ses propres sessions de console. AWS La ressource de politique représente la

AWS session de l'appelant. `arn:aws:sts::account-id:self` Le segment `account-id` ARN peut être remplacé par un caractère générique * dans les cas où la même politique d'autorisation est déployée sur plusieurs comptes, par exemple lorsque cette politique est déployée à l'aide des ensembles d'autorisations IAM Identity Center.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:SetContext",
      "Resource": "arn:aws:sts::account-id:self"
    }
  ]
}
```

Gérer AWS STS dans un Région AWS

Par défaut, le AWS Security Token Service (AWS STS) est disponible en tant que service global, et toutes les AWS STS demandes sont envoyées à un seul point de terminaison à`https://sts.amazonaws.com`. AWS recommande d'utiliser des AWS STS points de terminaison régionaux plutôt que des points de terminaison globaux pour réduire la latence, renforcer la redondance et augmenter la validité des jetons de session.

- Réduisez la latence : en passant vos AWS STS appels vers un point de terminaison géographiquement plus proche de vos services et applications, vous pouvez accéder à AWS STS des services avec une latence plus faible et de meilleurs temps de réponse.
- Intégrer de la redondance : vous pouvez limiter les effets d'une défaillance au sein d'une charge de travail à un nombre limité de composants avec une portée prévisible de maîtrise des impacts. L'utilisation de AWS STS points de terminaison régionaux vous permet d'aligner la portée de vos composants sur celle de vos jetons de session. Pour plus d'informations sur ce pilier de fiabilité, consultez Utilisation de l'isolation des défaillances pour protéger votre charge de travail dans le cadre AWS Well-Architected.
- Augmenter la validité des jetons de session — Les jetons de session provenant des AWS STS points de terminaison régionaux sont valides dans tous les Régions AWS cas. Les jetons de session provenant du point de terminaison STS global ne sont valides Régions AWS que s'ils sont activés par défaut. Si vous avez l'intention d'activer une nouvelle région pour votre compte, vous pouvez utiliser des jetons de session provenant des AWS STS points de terminaison régionaux.

Si vous choisissez d'utiliser le point de terminaison global, vous devez modifier la compatibilité régionale des jetons de AWS STS session pour le point de terminaison global. Cela garantit que les jetons sont valides dans tous les cas Régions AWS.

Gestion des jetons de session de point de terminaison global

La plupart Régions AWS sont activées pour toutes les opérations Services AWS par défaut. Ces régions sont automatiquement activées pour être utilisées avec AWS STS. Certaines régions, telles que l'Asie-Pacifique (Hong Kong), doivent être activées manuellement. Pour en savoir plus sur l'activation et la désactivation Régions AWS, voir Spécifier les comptes que Régions AWS votre compte peut utiliser dans le Guide de AWS Account Management référence. Lorsque vous activez ces AWS régions, elles sont automatiquement activées pour être utilisées avec AWS STS. Vous ne pouvez pas activer le AWS STS point de terminaison pour une région désactivée. Les jetons de session valides dans tous les domaines Régions AWS incluent plus de caractères que les jetons valides dans les régions activées par défaut. La modification de ce paramètre peut avoir un impact sur les systèmes existants où vous stockez temporairement les jetons.

Vous pouvez modifier ce paramètre à l'aide de l' AWS API AWS Management Console AWS CLI, ou.

Pour modifier la compatibilité régionale des jetons de session pour le point de terminaison global (console)

1. Connectez-vous en tant qu'utilisateur root ou utilisateur IAM avec les autorisations pour effectuer des tâches d'administration d'IAM. Pour modifier la compatibilité des jetons de session, vous devez posséder une politique qui autorise l'action `iam:SetSecurityTokenServicePreferences`.
2. Ouvrez la console IAM. Dans le panneau de navigation, choisissez Paramètres du compte.
3. Dans la section Security Token Service (STS), Jetons de session provenant des points de terminaison STS. Le point de terminaison global indique `Valid only in Régions AWS enabled by default`. Choisissez Change (Modifier).
4. Dans la boîte de dialogue Modifier la compatibilité des régions, sélectionnez Tout Régions AWS. Ensuite, choisissez Enregistrer les modifications.

> **Note**
>
> Les jetons de session valides dans tous les domaines Région AWS incluent plus de caractères que les jetons valides dans les régions activées par défaut. La modification

de ce paramètre peut avoir un impact sur les systèmes existants où vous stockez temporairement les jetons.

Pour modifier la compatibilité régionale des jetons de session pour le point de terminaison global (AWS CLI)

Définissez la version du jeton de session. Les jetons de version 1 ne sont valides Régions AWS que s'ils sont disponibles par défaut. Ces jetons ne fonctionnent pas dans les régions activées manuellement, par exemple, Asie-Pacifique (Hong Kong). Les jetons de la version 2 sont valides dans toutes les régions. Toutefois, les jetons de version 2 incluent plus de caractères et peuvent avoir un impact sur les systèmes où vous stockez temporairement les jetons.

- `aws iam set-security-token-service-preferences`

Pour modifier la compatibilité régionale des jetons de session pour le point de terminaison global (API AWS)

Définissez la version du jeton de session. Les jetons de version 1 ne sont valides Régions AWS que s'ils sont disponibles par défaut. Ces jetons ne fonctionnent pas dans les régions activées manuellement, par exemple, Asie-Pacifique (Hong Kong). Les jetons de la version 2 sont valides dans toutes les régions. Toutefois, les jetons de version 2 incluent plus de caractères et peuvent avoir un impact sur les systèmes où vous stockez temporairement les jetons.

- `SetSecurityTokenServicePreferences`

Activation et désactivation AWS STS dans un Région AWS

Lorsque vous activez les points de terminaison STS pour une région, vous AWS STS pouvez délivrer des informations d'identification temporaires aux utilisateurs et aux rôles de votre compte qui font une AWS STS demande. Ces informations d'identification peuvent ensuite être utilisées dans n'importe quelle région activée par défaut ou activée manuellement. Pour les régions activées par défaut, vous devez activer le point de terminaison STS régional sur le compte où les informations d'identification temporaires sont générées. Peu importe si un utilisateur est connecté au même compte ou à un compte différent lorsqu'il fait la demande. Pour les régions activées manuellement, vous devez activer la région à la fois sur le compte qui effectue la demande et sur le compte où les informations d'identification temporaires sont générées.

Par exemple, imaginez qu'un utilisateur du compte A souhaite envoyer une demande d'`sts:AssumeRole`API au point de terminaison AWS STS régional`https://sts.ap-east-1.amazonaws.com`. La demande concerne les informations d'identification temporaires du rôle nommé `Developer` dans le compte B. Étant donné qu'il s'agit d'une demande de création d'informations d'identification pour une entité du compte B, ce compte doit activer la région `ap-east-1`. Les utilisateurs du compte A (ou de tout autre compte) peuvent appeler le point de terminaison `ap-east-1` AWS STS pour demander les informations d'identification pour le compte B, que la région soit activée ou non dans leurs comptes.

> **Note**
>
> Les régions actives sont disponibles pour chaque personne qui utilise des informations d'identification temporaires de ce compte. Pour contrôler les utilisateurs ou rôles IAM qui peuvent accéder à la région, utilisez la clé de condition `aws:RequestedRegion` dans vos politiques d'autorisations.

Pour activer ou désactiver AWS STS dans une région activée par défaut (console)

1. Connectez-vous en tant qu'utilisateur root ou utilisateur IAM avec les autorisations pour effectuer des tâches d'administration d'IAM.
2. Ouvrez la console IAM, puis dans le panneau de navigation, sélectionnez Account settings (Paramètres du compte).
3. Dans la section Security Token Service (STS) Points de terminaison, recherchez la région que vous souhaitez configurer, puis choisissez Active ou Inactive dans la colonne d'état STS.
4. Dans la boîte de dialogue qui s'ouvre, choisissez Activate ou Deactivate (Activer ou Désactiver).

Pour les régions qui doivent être activées, nous les activons AWS STS automatiquement lorsque vous activez la région. Une fois que vous avez activé une région, elle AWS STS est toujours active pour cette région et vous ne pouvez pas la désactiver. Pour en savoir plus sur l'activation des régions désactivées par défaut, consultez la section Spécifier les régions que Régions AWS votre compte peut utiliser dans le Guide de AWS Account Management référence.

Écriture de code pour l'utilisation de AWS STS

Après avoir activé une région, vous pouvez diriger les appels d' AWS STS API vers cette région. L'extrait de code Java suivant montre comment configurer un `AWSSecurityTokenService` objet pour envoyer des demandes à la région Europe (Milan) (eu-south-1).

```
EndpointConfiguration regionEndpointConfig = new EndpointConfiguration("https://sts.eu-
south-1.amazonaws.com", "eu-south-1");
AWSSecurityTokenService stsRegionalClient =
 AWSSecurityTokenServiceClientBuilder.standard()
.withCredentials(credentials)
.withEndpointConfiguration(regionEndpointConfig)
.build();
```

AWS STS vous recommande de passer des appels vers un point de terminaison régional. Pour savoir comment activer manuellement une région, voir Spécifier les régions que Régions AWS votre compte peut utiliser dans le Guide de AWS Account Management référence.

Dans l'exemple, la première ligne instancie un objet `EndpointConfiguration` appelé `regionEndpointConfig`, en transmettant l'URL du point de terminaison et la Région AWS comme paramètres.

Pour savoir comment définir des points de terminaison AWS STS régionaux à l'aide d'une variable d'environnement pour les AWS SDK, consultez la section Points de terminaison AWS STS régionalisés dans le Guide de référence AWS des SDK et des outils.

Pour toutes les autres combinaisons de langage et d'environnement de programmation, reportez-vous à la documentation du kit SDK approprié.

Régions et points de terminaison

Le tableau suivant répertorie les régions et leurs points de terminaison. Il indique les régions activées par défaut et celles que vous pouvez activer ou désactiver.

Nom de la région	Point de terminaison	Activé par défaut	Activer/désactiver manuellement
--Global--	sts.amazonaws.com	 Oui	 Non
USA Est (Ohio)	sts.us-east-2.amazonaws.com	 Oui	 Oui
USA Est (Virginie du Nord)	sts.us-east-1.amazonaws.com	 Oui	 Non
USA Ouest (Californie du Nord)	sts.us-west-1.amazonaws.com	 Oui	 Oui
USA Ouest (Oregon)	sts.us-west-2.amazonaws.com	 Oui	 Oui
Afrique (Le Cap)	sts.af-south-1.amazonaws.com	 Non[1]	 Non

Nom de la région	Point de terminaison	Activé par défaut	Activer/désactiver manuellement
Asie-Pacifique (Hong Kong)	sts.ap-east-1.amazonaws.com	 Non[1]	 Non
Asie-Pacifique (Hyderabad)	sts.ap-south-2.amazonaws.com	 Non[1]	 Non
Asie-Pacifique (Jakarta)	sts.ap-southeast-3.amazonaws.com	 Non[1]	 Non
Asie-Pacifique (Melbourne)	sts.ap-southeast-4.amazonaws.com	 Non[1]	 Non
Asie-Pacifique (Mumbai)	sts.ap-south-1.amazonaws.com	 Oui	 Oui
Asie-Pacifique (Osaka)	sts.ap-northeast-3.amazonaws.com	 Oui	 Oui

Nom de la région	Point de terminaison	Activé par défaut	Activer/désactiver manuellement
Asie-Pacifique (Séoul)	sts.ap-northeast-2.amazonaws.com	 Oui	 Oui
Asie-Pacifique (Singapour)	sts.ap-southeast-1.amazonaws.com	 Oui	 Oui
Asie-Pacifique (Sydney)	sts.ap-southeast-2.amazonaws.com	 Oui	 Oui
Asie-Pacifique (Tokyo)	sts.ap-northeast-1.amazonaws.com	 Oui	 Oui
Canada (Centre)	sts.ca-central-1.amazonaws.com	 Oui	 Oui
Canada Ouest (Calgary)	sts.ca-west-1.amazonaws.com	 Oui	 Oui

Nom de la région	Point de terminaison	Activé par défaut	Activer/désactiver manuellement
Chine (Beijing)	sts.cn-north-1.amazonaws.com.cn	 Oui[1]	 Non
Chine (Ningxia)	sts.cn-northwest-1.amazonaws.com.cn	 Oui[1]	 Oui
Europe (Francfort)	sts.eu-central-1.amazonaws.com	 Oui	 Oui
Europe (Irlande)	sts.eu-west-1.amazonaws.com	 Oui	 Oui
Europe (Londres)	sts.eu-west-2.amazonaws.com	 Oui	 Oui
Europe (Milan)	sts.eu-south-1.amazonaws.com	 Non[1]	 Non

Nom de la région	Point de terminaison	Activé par défaut	Activer/désactiver manuellement
Europe (Paris)	sts.eu-west-3.amazonaws.com	 Oui	 Oui
Europe (Espagne)	sts.eu-south-2.amazonaws.com	 Non[1]	 Non
Europe (Stockholm)	sts.eu-north-1.amazonaws.com	 Oui	 Oui
Europe (Zurich)	sts.eu-central-2.amazonaws.com	 Non[1]	 Non
Israël (Tel Aviv)	sts.il-central-1.amazonaws.com	 Non[1]	 Non
Moyen-Orient (Bahreïn)	sts.me-south-1.amazonaws.com	 Non[1]	 Non

Nom de la région	Point de terminaison	Activé par défaut	Activer/désactiver manuellement
Moyen-Orient (EAU)	sts.me-central-1.amazonaws.com	 Non[1]	 Non
Amérique du Sud (São Paulo)	sts.sa-east-1.amazonaws.com	 Oui	 Oui

[1]Vous devez activer la région pour l'utiliser. Cela active automatiquement AWS STS. Vous ne pouvez pas activer ou désactiver manuellement AWS STS dans ces régions.

[2]Pour l'utiliser AWS en Chine, vous avez besoin d'un compte et d'informations d'identification spécifiques AWS à la Chine.

AWS CloudTrail et points de terminaison régionaux

Les appels vers les points de terminaison régionaux et globaux sont enregistrés dans le champ `tlsDetails` dans AWS CloudTrail. Les appels vers les points de terminaison régionaux`us-east-2.amazonaws.com`, tels que, sont connectés CloudTrail à la région appropriée. Les appels vers le point de terminaison global, `sts.amazonaws.com`, sont consignés en tant qu'appels à un service international. Les événements relatifs aux AWS STS points de terminaison globaux sont enregistrés dans us-east-1.

> **Note**
>
> `tlsDetails` peut uniquement être consulté pour les services qui prennent en charge ce champ. Voir les informations sur les services compatibles avec le protocole TLS CloudTrail dans le guide de l'AWS CloudTrail utilisateur
> Pour plus d'informations, voir Journalisation des appels IAM et AWS STS API avec AWS CloudTrail.

Utilisation des jetons porteurs

Certains AWS services nécessitent que vous soyez autorisé à obtenir un jeton de support de AWS STS service avant de pouvoir accéder à leurs ressources par programmation. Ces services prennent en charge un protocole qui vous oblige à utiliser un jeton porteur au lieu d'utiliser une demande signée traditionnelle Signature Version 4. Lorsque vous effectuez des opérations AWS CLI ou des opérations d' AWS API qui nécessitent des jetons porteurs, le AWS service demande un jeton porteur en votre nom. Le service vous fournit le jeton, que vous pouvez ensuite utiliser pour effectuer toute opération ultérieure dans ce service.

AWS STS les jetons de support incluent des informations issues de votre authentification principale d'origine qui peuvent affecter vos autorisations. Ces informations peuvent comprendre des balises de principal ou de session, ainsi que des politiques de session. L'ID de clé d'accès du jeton commence par le préfixe `ABIA`. Cela vous permet d'identifier les opérations effectuées à l'aide de jetons de support dans vos CloudTrail journaux.

> **Important**
>
> Le jeton porteur ne peut être utilisé que pour les appels vers le service qui le génère et dans la région dans laquelle il a été généré. Vous ne pouvez pas l'utiliser pour effectuer des opérations dans d'autres services ou régions.

Un exemple de service qui prend en charge les jetons au porteur est AWS CodeArtifact. Avant de pouvoir interagir avec AWS CodeArtifact un gestionnaire de packages tel que NPM, Maven ou PIP, vous devez appeler l'opération. `aws codeartifact get-authorization-token` Cette opération renvoie un jeton porteur que vous pouvez utiliser pour effectuer des AWS CodeArtifact opérations. Si vous préférez, vous pouvez aussi utiliser la commande `aws codeartifact login` qui exécute la même opération, puis configure automatiquement votre client.

Si vous effectuez une action dans un AWS service qui génère un jeton porteur pour vous, vous devez disposer des autorisations suivantes dans votre politique IAM :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowServiceBearerToken",
            "Effect": "Allow",
```

```
            "Action": "sts:GetServiceBearerToken",
            "Resource": "*"
        }
    ]
}
```

Pour obtenir un exemple de jeton de support de service, consultezUtilisation de stratégies basées sur l'identité pour AWS CodeArtifact dans le Guide de l'utilisateur AWS CodeArtifact.

Exemples d'application qui utilisent des informations d'identification temporaires

Vous pouvez utiliser AWS Security Token Service (AWS STS) pour créer et fournir à des utilisateurs de confiance des informations d'identification de sécurité temporaires qui peuvent contrôler l'accès à vos AWS ressources. Pour plus d'informations sur AWS STS, consultez Informations d'identification de sécurité temporaires dans IAM. Pour découvrir comment gérer les informations d'identification AWS STS de sécurité temporaires, vous pouvez télécharger les exemples d'applications suivants qui mettent en œuvre des exemples de scénarios complets :

- Activation de la fédération pour AWS utiliser Windows Active Directory, ADFS et SAML 2.0. Montre comment déléguer l'accès à l'aide de la fédération d'entreprise pour AWS à l'aide de Windows Active Directory (AD), d'Active Directory Federation Services (Services de fédération Active Directory - ADFS) 2.0 et de SAML (Security Assertion Markup Language - Langage de balisage d'assertion de sécurité) 2.0.
- Activation de l'accès à la AWS console par un courtier d'identité personnaliséExplique comment créer un proxy de fédération permettant d'autoriser l'authentification unique (SSO) afin que les utilisateurs Active Directory existants puissent se connecter à AWS Management Console.
- Comment utiliser Shibboleth pour l'authentification unique au. AWS Management Console . Explique comment utiliser Shibboleth et SAML pour fournir aux utilisateurs un accès avec authentification unique (SSO) à l' AWS Management Console.

Exemples pour la fédération OIDC

Les exemples d'applications suivants montrent comment utiliser OIDCFederation avec des fournisseurs tels que Login with Amazon, Amazon Cognito, Facebook ou Google. Vous pouvez échanger l'authentification de ces fournisseurs contre des informations AWS de sécurité temporaires pour accéder aux AWS services.

- Tutoriels Amazon Cognito — Nous vous recommandons d'utiliser Amazon Cognito avec les SDK pour AWS le développement mobile. Amazon Cognito offre le moyen le plus simple de gérer l'identité pour les applications mobiles et fournit des fonctions supplémentaires telles que la synchronisation et l'identité entre périphériques. Pour plus d'informations sur Amazon Cognito, consultez Authentification avec Amplify dans la documentation Amplify.

Activation de l'accès à la AWS console par un courtier d'identité personnalisé

Vous pouvez écrire et exécuter du code afin de créer une URL qui permet aux utilisateurs qui se connectent au réseau de votre organisation d'accéder en toute sécurité à AWS Management Console. L'URL inclut un jeton de connexion que vous obtenez AWS et qui authentifie l'utilisateur. AWS La session de console obtenue peut inclure un `AccessKeyId` distinct en raison de la fédération. Pour suivre l'utilisation des clés d'accès pour la connexion à la fédération par le biais d' CloudTrail événements connexes, consultez la section Événements Journalisation des appels IAM et AWS STS API avec AWS CloudTrail de AWS Management Console connexion.

> **Note**
>
> Si votre organisation utilise un fournisseur d'identité (IdP) qui est compatible avec SAML, vous pouvez configurer l'accès à la console sans écrire de code. Cela fonctionne avec des fournisseurs tels que Microsoft Active Directory Federation Services ou Shibboleth (open source). Pour plus de détails, consultez Permettre aux utilisateurs fédérés SAML 2.0 d'accéder au AWS Management Console.

Pour permettre aux utilisateurs de votre organisation d'accéder au AWS Management Console, vous pouvez créer un courtier d'identité personnalisé qui exécute les étapes suivantes :

1. Vérifier que l'utilisateur est authentifié par votre système d'identité local.
2. Appelez les opérations AWS Security Token Service (AWS STS) AssumeRole(recommandé) ou GetFederationTokenAPI pour obtenir des informations d'identification de sécurité temporaires pour l'utilisateur. Pour connaître les différentes méthodes que vous pouvez utiliser pour endosser un rôle, consultez Utilisation de rôles IAM. Pour savoir comment transmettre des balises de session facultatives lorsque vous obtenez vos informations d'identification de sécurité, veuillez consulter Transmission des balises de session AWS STS.

- Si vous utilisez l'une des opérations d'API `AssumeRole*` pour obtenir les informations d'identification de sécurité temporaires pour un rôle, vous pouvez inclure le paramètre `DurationSeconds` dans votre appel. Ce paramètre spécifie la durée de la session de votre rôle, entre 900 secondes (15 minutes) et la durée de session maximale pour le rôle. Lorsque vous utilisez `DurationSeconds` dans une opération `AssumeRole*`, vous devez l'appeler en tant qu'utilisateur IAM avec des informations d'identification à long terme. Sinon, l'appel au point de terminaison de fédération de l'étape 3 échoue. Pour savoir comment afficher ou modifier la valeur maximale pour un rôle, consultez Affichage du paramètre de durée de session maximale pour un rôle.
- Si vous utilisez l'opération d'API `GetFederationToken` pour obtenir les informations d'identification, vous pouvez inclure le paramètre `DurationSeconds` dans votre appel. Ce paramètre spécifie la durée de votre session de rôle. La valeur peut être comprise entre 900 secondes (15 minutes) et 129 600 secondes (36 heures). Vous pouvez effectuer cet appel d'API uniquement en utilisant les informations de AWS sécurité à long terme d'un utilisateur IAM. Vous pouvez également effectuer ces appels à l'aide Utilisateur racine d'un compte AWS d'informations d'identification, mais nous ne le recommandons pas. Si vous effectuez cet appel en tant qu'utilisateur racine, la session par défaut dure une heure. Ou vous pouvez spécifier une session de 900 secondes (15 minutes) à 3 600 secondes (une heure).

3. Appelez le point de terminaison de la AWS fédération et fournissez les informations d'identification de sécurité temporaires pour demander un jeton de connexion.
4. Créer une URL incluant le jeton pour la console :
   - Si vous utilisez l'une des opérations d'API `AssumeRole*` dans votre URL, vous pouvez inclure le paramètre HTTP `SessionDuration`. Ce paramètre spécifie la durée de la session de la console, de 900 secondes (15 minutes) à 43 200 secondes (12 heures).
   - Si vous utilisez l'opération d'API `GetFederationToken` dans votre URL, vous pouvez inclure le paramètre `DurationSeconds`. Ce paramètre spécifie la durée de la session de console fédérée. La valeur peut être comprise entre 900 secondes (15 minutes) et 129 600 secondes (36 heures).

   > **Note**
   >
   > - N'utilisez pas le paramètre HTTP `SessionDuration` si vous avez obtenu les informations d'identification avec `GetFederationToken`. L'opération échouerait.
   > - L'utilisation des informations d'identification pour qu'un rôle endosse un rôle différent est appelée chaînes de rôles. Lorsque vous utilisez la création de chaînes de rôles,

vos nouvelles informations d'identification sont limitées à une durée maximale d'une heure. Lorsque vous utilisez des rôles pour accorder des autorisations aux applications qui s'exécutent sur des instances EC2, ces applications ne sont pas soumises à cette limitation.

5. Donner l'URL à l'utilisateur ou appeler l'URL pour le compte de l'utilisateur.

L'URL fournie par le point de terminaison de fédération est valide pendant 15 minutes à compter de sa création. Cette durée est différente de celle (en secondes) de la session des informations d'identification de sécurité temporaires associées à l'URL. Ces informations d'identification sont valides pendant la durée spécifiée lors de leur création et ce, à compter du moment où elles sont créées.

Important

L'URL donne accès à vos AWS ressources via les autorisations AWS Management Console si vous avez activé les autorisations dans les informations d'identification de sécurité temporaires associées. Pour cette raison, vous devez considérer l'URL comme un secret. Nous vous recommandons de retourner l'URL via une redirection sécurisée, par exemple, en utilisant un code de statut de réponse HTTP 302 via une connexion SSL. Pour plus d'informations sur le code de statut de réponse HTTP 302, consultez RFC 2616, section 10.3.3.

Pour effectuer ces tâches, vous pouvez utiliser l'API Query HTTPS pour AWS Identity and Access Management (IAM) et AWS Security Token Service (AWS STS). Vous pouvez également utiliser des langages de programmation tels que Java, Ruby ou C#, avec le kit SDK AWS approprié. Chacune de ces méthodes est décrite dans les rubriques suivantes.

Rubriques

- Exemple de code utilisant les opérations d'API Query IAM
- Exemple de code utilisant Python
- Exemple de code utilisant Java
- Exemple montrant comment créer l'URL (Ruby)

Exemple de code utilisant les opérations d'API Query IAM

Vous pouvez créer une URL qui donne à vos utilisateurs fédérés un accès direct au AWS Management Console. Cette tâche utilise les API de requête IAM et AWS STS HTTPS. Pour de plus amples informations sur les demandes de requête, veuillez consulter Envoi de demandes de requête.

> **Note**
>
> La procédure suivante contient des exemples de chaînes de texte. Pour faciliter la lecture, des sauts de ligne ont été ajoutés aux exemples les plus longs. Lorsque vous créez ces chaînes pour votre propre utilisation, il convient d'omettre les sauts de ligne.

Pour donner à un utilisateur fédéré l'accès à vos ressources depuis le AWS Management Console

1. Authentifiez l'utilisateur dans votre système d'identité et d'autorisation.
2. Obtenez des informations d'identification de sécurité temporaires pour l'utilisateur. Les informations d'identification temporaires incluent un ID de clé d'accès, une clé d'accès secrète et un jeton (token) de session. Pour plus d'informations sur la création d'informations d'identification temporaires, consultez Informations d'identification de sécurité temporaires dans IAM.

   Pour obtenir des informations d'identification temporaires, vous devez appeler l' AWS STS AssumeRoleAPI (recommandé) ou l'GetFederationTokenAPI. Pour plus d'informations sur les différences entre ces opérations d'API, consultez la section Comprendre les options d'API pour déléguer l'accès à votre AWS compte en toute sécurité dans le blog sur la AWS sécurité.

   > **Important**
   >
   > Lorsque vous utilisez l'GetFederationTokenAPI pour créer des informations d'identification de sécurité temporaires, vous devez spécifier les autorisations que les informations d'identification accordent à l'utilisateur qui assume le rôle. Dès lors que les opérations d'API commencent par `AssumeRole*`, vous utilisez un rôle IAM pour accorder les autorisations. Pour les autres opérations d'API, le mécanisme varie en fonction de l'API. Pour en savoir plus, consultez Contrôle des autorisations affectées aux informations d'identification de sécurité temporaires. De plus, si vous utilisez les opérations d'API `AssumeRole*`, vous devez les appeler en tant qu'utilisateur IAM avec

des informations d'identification à long terme. Sinon, l'appel au point de terminaison de fédération de l'étape 3 échoue.

3. Une fois que vous avez obtenu les informations d'identification de sécurité temporaires, insérez-les dans une chaîne de session JSON pour les échanger contre un jeton de connexion. L'exemple suivant montre comment encoder les informations d'identification. Vous remplacez le texte de l'espace réservé par les valeurs appropriées dans les informations d'identification reçues à l'étape précédente.

```
{"sessionId":"*** temporary access key ID ***",
"sessionKey":"*** temporary secret access key ***",
"sessionToken":"*** session token ***"}
```

4. Encodez par URL la chaîne de session de l'étape précédente. Dans la mesure où les informations que vous encodez sont sensibles, nous vous recommandons de ne pas utiliser de service web pour cette opération. Utilisez, à la place, une fonction installée en local ou une fonctionnalité de votre boîte à outils de développement pour encoder cette information de manière sécurisée. Vous pouvez utiliser la fonction `urllib.quote_plus` dans Python, la fonction `URLEncoder.encode` dans Java ou la fonction `CGI.escape` dans Ruby. Consultez les exemples dans la suite de cette rubrique.

5.

> **Note**
>
> AWS prend en charge les requêtes POST ici.

Envoyez votre demande au point de terminaison AWS de la fédération :

`https://region-code.signin.aws.amazon.com/federation`

Pour obtenir la liste des valeurs possibles de *region-code*, consultez la colonne Region (Région) des AWS Sign-In endpoints (Points de terminaison de connexion). Vous pouvez éventuellement utiliser le point de terminaison de fédération de AWS connexion par défaut :

`https://signin.aws.amazon.com/federation`

La demande doit inclure les paramètres `Action` et `Session`, et (éventuellement), si vous avez utilisé une opération d'API `AssumeRole*`, un paramètre HTTP `SessionDuration` comme illustré dans l'exemple suivant.

```
Action = getSigninToken
SessionDuration = time in seconds
Session = *** the URL encoded JSON string created in steps 3 & 4 ***
```

> **Note**
>
> Les instructions suivantes de cette étape ne fonctionnent qu'avec les requêtes GET.

Ce paramètre HTTP `SessionDuration` spécifie la durée de la session de console. Cette durée est différente de celle des informations d'identification temporaires que vous spécifiez à l'aide du paramètre `DurationSeconds`. Vous pouvez spécifier une valeur `SessionDuration` maximale de 43 200 (12 heures). Si le `SessionDuration` paramètre est absent, la session utilise par défaut la durée des informations d'identification que vous avez récupérées AWS STS à l'étape 2 (qui est par défaut d'une heure). Consultez la documentation de l'API `AssumeRole` pour plus de détails sur la spécification d'une durée à l'aide du paramètre `DurationSeconds`. La possibilité de créer une session de console d'une durée supérieure à une heure fait partie intégrante de l'opération `getSigninToken` du point de terminaison de fédération.

> **Note**
>
> - N'utilisez pas le paramètre HTTP `SessionDuration` si vous avez obtenu les informations d'identification avec `GetFederationToken`. L'opération échouerait.
> - L'utilisation des informations d'identification pour qu'un rôle endosse un rôle différent est appelée chaînes de rôles. Lorsque vous utilisez la création de chaînes de rôles, vos nouvelles informations d'identification sont limitées à une durée maximale d'une heure. Lorsque vous utilisez des rôles pour accorder des autorisations aux applications qui s'exécutent sur des instances EC2, ces applications ne sont pas soumises à cette limitation.

Lorsque vous activez des sessions de console avec une durée prolongée, vous augmentez le risque d'exposition aux informations d'identification. Pour vous aider à réduire ce risque, vous pouvez immédiatement désactiver les sessions de console actives pour un rôle en choisissant Révoquer les sessions sur la page Résumé du rôle dans la console IAM. Pour

plus d'informations, veuillez consulter Révocation des informations d'identification de sécurité temporaires d'un rôle IAM.

Votre demande devrait ressembler à l'exemple suivant : Des retours à la ligne ont été ajoutés ici pour faciliter la lecture, mais vous devez envoyer une chaîne composée d'une seule ligne.

```
https://signin.aws.amazon.com/federation
?Action=getSigninToken
&SessionDuration=1800
&Session=%7B%22sessionId%22%3A+%22ASIAJUMHIZPTOKTBMK5A%22%2C+%22sessionKey%22
%3A+%22LSD7LWI%2FL%2FN%2BgYpan5QFz0XUpc8s7HYjRsgcsrsm%22%2C+%22sessionToken%2
2%3A+%22FQoDYXdzEBQaDLbj3VWv2u50NN%2F3yyLSASwYtWhPnGPMNmzZFfZsL0Qd3vtYHw5A5dW
AjOsrkdPkghomIe3mJip5%2F0djDBbo7SmO%2FENDEiCdpsQKodTpleKA8xQq0CwFg6a69xdEBQT8
FipATnLbKoyS4b%2FebhnsTUjZZQWp0wXXqFF7gSm%2FMe2tXe0jzsdP0O12obez9lijPSdF1k2b5
PfGhiuyAR9aD5%2BubM0pY86fKex1qsytjvyTbZ9nXe6DvxVDcnCOhOGETJ7XFkSFdH0v%2FYR25C
UAhJ3nXIkIbG7Ucv9cOEpCf%2Fg23ijRgILIBQ%3D%3D%22%7D
```

La réponse du point de terminaison de fédération est un document JSON avec une valeur `SigninToken`. Elle sera similaire à l'exemple suivant.

```
{"SigninToken":"*** the SigninToken string ***"}
```

6.

> **Note**
>
> AWS prend en charge les requêtes POST ici.

Enfin, créez l'URL que vos utilisateurs fédérés utiliseront pour accéder à AWS Management Console. L'URL est la même URL de point de terminaison de fédération que celle que vous avez utilisée dans Step 5, à laquelle sont ajoutés les paramètres suivants :

```
?Action = login
&Issuer = *** the form-urlencoded URL for your internal sign-in page ***
&Destination = *** the form-urlencoded URL to the desired AWS console page ***
&SigninToken = *** the value of SigninToken received in the previous step ***
```

> **Note**
>
> Les instructions suivantes de cette étape ne fonctionnent qu'avec l'API GET.

L'exemple suivant montre comment se présentera l'URL finale. L'URL est valide pendant 15 minutes à partir du moment où elle est créée. Les informations d'identification de sécurité temporaires et la session de console intégrées dans l'URL sont valides pendant la durée spécifiée dans le paramètre HTTP `SessionDuration` lors de leur demande initiale.

```
https://signin.aws.amazon.com/federation
?Action=login
&Issuer=https%3A%2F%2Fexample.com
&Destination=https%3A%2F%2Fconsole.aws.amazon.com%2F
&SigninToken=VCQgs5qZZt3Q6fn8Tr5EXAMPLEmLnwB7JjUc-SHwnUUWabcRdnWsi4DBn-dvC
CZ85wrD0nmldUcZEXAMPLE-vXYH4Q__mleuF_W2BE5HYexbe9y4Of-kje53SsjNNecATfjIzpW1
WibbnH6YcYRiBoffZBGExbEXAMPLE5aiKX4THWjQKC6gg6alHu6JFrnOJoK3dtP6I9a6hi6yPgm
iOkPZMmNGmhsvVxetKzr8mx3pxhHbMEXAMPLETv1pij0rok3IyCR2YVcIjqwfWv32HU2Xlj471u
3fU6uOfUComeKiqTGX974xzJOZbdmX_t_lLrhEXAMPLEDDIisSnyHGw2xaZZqudm4mo2uTDk9Pv
9l5K0ZCqIgEXAMPLEcA6tgLPykEWGUyH6BdSC6166n4M4JkXIQgac7_7821YqixsNxZ6rsrpzwf
nQoS14O7R0eJCCJ684EXAMPLEZRdBNnuLbUYpz2Iw3vIN0tQgOujwnwydPscM9F7foaEK3jwMkg
Apeb1-6L_OB12MZhuFxx55555EXAMPLEhyETEd4ZulKPdXHkgl6T9ZkIlHz2Uy1RUTUhhUxNtSQ
nWc5xkbBoEcXqpoSIeK7yhje9Vzhd61AEXAMPLElbWeouACEMG6-Vd3dAgFYd6i5FYoyFrZLWvm
0LSG7RyYKeYN5VIzUk3YWQpyjP0RiT5KUrsUi-NEXAMPLExMOMdoODBEgKQsk-iu2ozh6r8bxwC
RNhujg
```

Exemple de code utilisant Python

L'exemple suivant montre comment utiliser Python pour créer par programmation une URL qui donne aux utilisateurs fédérés un accès direct à AWS Management Console. Voici deux exemples :

- Fédérez via des requêtes GET pour AWS
- Fédérez via des requêtes POST pour AWS

Les deux exemples utilisent l'AssumeRoleAPI AWS SDK for Python (Boto3)and pour obtenir des informations d'identification de sécurité temporaires.

Utiliser des requêtes GET

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'
```

```
# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your Compte AWS,
# call "AssumeRole" to get temporary access keys for the federated user

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,
# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleSession",
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = "?Action=getSigninToken"
request_parameters += "&SessionDuration=43200"
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)
```

```
request_parameters += "&Session=" +
 quote_plus_function(json_string_with_temp_credentials)
request_url = "https://signin.aws.amazon.com/federation" + request_parameters
r = requests.get(request_url)
# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create URL where users can use the sign-in token to sign in to
# the console. This URL must be used within 15 minutes after the
# sign-in token was issued.
request_parameters = "?Action=login"
request_parameters += "&Issuer=Example.org"
request_parameters += "&Destination=" + quote_plus_function("https://
console.aws.amazon.com/")
request_parameters += "&SigninToken=" + signin_token["SigninToken"]
request_url = "https://signin.aws.amazon.com/federation" + request_parameters

# Send final URL to stdout
print (request_url)
```

Utiliser des requêtes POST

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'
import os
from selenium import webdriver # 'pip install selenium', 'brew install chromedriver'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your A Compte AWS,
# call "AssumeRole" to get temporary access keys for the federated user

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,

# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
if sys.version_info[0] < 3:
```

```
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)

sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleDemoSession",
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = {}
request_parameters['Action'] = 'getSigninToken'
request_parameters['SessionDuration'] = '43200'
request_parameters['Session'] = json_string_with_temp_credentials

request_url = "https://signin.aws.amazon.com/federation"
r = requests.post( request_url, data=request_parameters)

# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create a POST request where users can use the sign-in token to sign in to
# the console. The POST request must be made within 15 minutes after the
# sign-in token was issued.
request_parameters = {}
request_parameters['Action'] = 'login'
```

```
request_parameters['Issuer']='Example.org'
request_parameters['Destination'] = 'https://console.aws.amazon.com/'
request_parameters['SigninToken'] =signin_token['SigninToken']

jsrequest = '''
var form = document.createElement('form');
form.method = 'POST';
form.action = '{request_url}';
request_parameters = {request_parameters}
for (var param in request_parameters) {{
    if (request_parameters.hasOwnProperty(param)) {{
        const hiddenField = document.createElement('input');
        hiddenField.type = 'hidden';
        hiddenField.name = param;
        hiddenField.value = request_parameters[param];
        form.appendChild(hiddenField);
    }}
}}
document.body.appendChild(form);
form.submit();
'''.format(request_url=request_url, request_parameters=request_parameters)

driver = webdriver.Chrome()
driver.execute_script(jsrequest);
```

Exemple de code utilisant Java

L'exemple suivant montre comment utiliser Java pour créer par programmation une URL qui donne aux utilisateurs fédérés un accès direct à AWS Management Console. L'extrait de code suivant utilise le kit SDK AWS pour Java.

```
import java.net.URLEncoder;
import java.net.URL;
import java.net.URLConnection;
import java.io.BufferedReader;
import java.io.InputStreamReader;
// Available at http://www.json.org/java/index.html
import org.json.JSONObject;
import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClient;
import com.amazonaws.services.securitytoken.model.Credentials;
import com.amazonaws.services.securitytoken.model.GetFederationTokenRequest;
```

```
import com.amazonaws.services.securitytoken.model.GetFederationTokenResult;


/* Calls to AWS STS API operations must be signed using the access key ID
   and secret access key of an IAM user or using existing temporary
   credentials. The credentials should not be embedded in code. For
   this example, the code looks for the credentials in a
   standard configuration file.
*/
AWSCredentials credentials =
  new PropertiesCredentials(
         AwsConsoleApp.class.getResourceAsStream("AwsCredentials.properties"));

AWSSecurityTokenServiceClient stsClient =
  new AWSSecurityTokenServiceClient(credentials);

GetFederationTokenRequest getFederationTokenRequest =
  new GetFederationTokenRequest();
getFederationTokenRequest.setDurationSeconds(1800);
getFederationTokenRequest.setName("UserName");

// A sample policy for accessing Amazon Simple Notification Service (Amazon SNS) in the
 console.

String policy = "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Action\":\"sns:*\"," +
  "\"Effect\":\"Allow\",\"Resource\":\"*\"}]}";

getFederationTokenRequest.setPolicy(policy);

GetFederationTokenResult federationTokenResult =
  stsClient.getFederationToken(getFederationTokenRequest);

Credentials federatedCredentials = federationTokenResult.getCredentials();

// The issuer parameter specifies your internal sign-in
// page, for example https://mysignin.internal.mycompany.com/.
// The console parameter specifies the URL to the destination console of the
// AWS Management Console. This example goes to Amazon SNS.
// The signin parameter is the URL to send the request to.

String issuerURL = "https://mysignin.internal.mycompany.com/";
String consoleURL = "https://console.aws.amazon.com/sns";
String signInURL = "https://signin.aws.amazon.com/federation";
```

```
// Create the sign-in token using temporary credentials,
// including the access key ID,  secret access key, and session token.
String sessionJson = String.format(
  "{\"%1$s\":\"%2$s\",\"%3$s\":\"%4$s\",\"%5$s\":\"%6$s\"}",
  "sessionId", federatedCredentials.getAccessKeyId(),
  "sessionKey", federatedCredentials.getSecretAccessKey(),
  "sessionToken", federatedCredentials.getSessionToken());

// Construct the sign-in request with the request sign-in token action, a
// 12-hour console session duration, and the JSON document with temporary
// credentials as parameters.

String getSigninTokenURL = signInURL +
                           "?Action=getSigninToken" +
                           "&DurationSeconds=43200" +
                           "&SessionType=json&Session=" +
                           URLEncoder.encode(sessionJson,"UTF-8");

URL url = new URL(getSigninTokenURL);

// Send the request to the AWS federation endpoint to get the sign-in token
URLConnection conn = url.openConnection ();

BufferedReader bufferReader = new BufferedReader(new
  InputStreamReader(conn.getInputStream()));
String returnContent = bufferReader.readLine();

String signinToken = new JSONObject(returnContent).getString("SigninToken");

String signinTokenParameter = "&SigninToken=" + URLEncoder.encode(signinToken,"UTF-8");

// The issuer parameter is optional, but recommended. Use it to direct users
// to your sign-in page when their session expires.

String issuerParameter = "&Issuer=" + URLEncoder.encode(issuerURL, "UTF-8");

// Finally, present the completed URL for the AWS console session to the user

String destinationParameter = "&Destination=" + URLEncoder.encode(consoleURL,"UTF-8");
String loginURL = signInURL + "?Action=login" +
                     signinTokenParameter + issuerParameter + destinationParameter;
```

Exemple montrant comment créer l'URL (Ruby)

L'exemple suivant montre comment utiliser Ruby pour créer par programmation une URL qui donne aux utilisateurs fédérés un accès direct à AWS Management Console. Cet extrait de code utilise le kit SDK AWS pour Ruby.

```
require 'rubygems'
require 'json'
require 'open-uri'
require 'cgi'
require 'aws-sdk'

# Create a new STS instance
#
# Note: Calls to AWS STS API operations must be signed using an access key ID
# and secret access key. The credentials can be in EC2 instance metadata
# or in environment variables and will be automatically discovered by
# the default credentials provider in the AWS Ruby SDK.
sts = Aws::STS::Client.new()

# The following call creates a temporary session that returns
# temporary security credentials and a session token.
# The policy grants permissions to work
# in the AWS SNS console.

session = sts.get_federation_token({
  duration_seconds: 1800,
  name: "UserName",
  policy: "{\"Version\":\"2012-10-17\",\"Statement\":{\"Effect\":\"Allow\",\"Action\":
\"sns:*\",\"Resource\":\"*\"}}",
})

# The issuer value is the URL where users are directed (such as
# to your internal sign-in page) when their session expires.
#
# The console value specifies the URL to the destination console.
# This example goes to the Amazon SNS console.
#
# The sign-in value is the URL of the AWS STS federation endpoint.
issuer_url = "https://mysignin.internal.mycompany.com/"
console_url = "https://console.aws.amazon.com/sns"
signin_url = "https://signin.aws.amazon.com/federation"
```

```
# Create a block of JSON that contains the temporary credentials
# (including the access key ID, secret access key, and session token).
session_json = {
  :sessionId => session.credentials[:access_key_id],
  :sessionKey => session.credentials[:secret_access_key],
  :sessionToken => session.credentials[:session_token]
}.to_json

# Call the federation endpoint, passing the parameters
# created earlier and the session information as a JSON block.
# The request returns a sign-in token that's valid for 15 minutes.
# Signing in to the console with the token creates a session
# that is valid for 12 hours.
get_signin_token_url = signin_url +
                       "?Action=getSigninToken" +
                       "&SessionType=json&Session=" +
                       CGI.escape(session_json)

returned_content = URI.parse(get_signin_token_url).read

# Extract the sign-in token from the information returned
# by the federation endpoint.
signin_token = JSON.parse(returned_content)['SigninToken']
signin_token_param = "&SigninToken=" + CGI.escape(signin_token)

# Create the URL to give to the user, which includes the
# sign-in token and the URL of the console to open.
# The "issuer" parameter is optional but recommended.
issuer_param = "&Issuer=" + CGI.escape(issuer_url)
destination_param = "&Destination=" + CGI.escape(console_url)
login_url = signin_url + "?Action=login" + signin_token_param +
  issuer_param + destination_param
```

Ressources supplémentaires pour les informations d'identification de sécurité temporaires

Les scénarios et applications suivants peuvent vous aider à utiliser les informations d'identification de sécurité temporaires :

- Comment s'intégrer AWS STS SourceIdentity à votre fournisseur d'identité. Cet article explique comment configurer l' AWS STS `SourceIdentity`attribut lorsque vous utilisez Okta, Ping ou OneLogin comme IdP.

- Fédération OIDC. Cette section explique comment configurer les rôles IAM lorsque vous utilisez la fédération OIDC et l'`AssumeRoleWithWebIdentity`API.
- Configuration de l'accès aux API protégé par MFA. Cette rubrique explique comment utiliser les rôles pour exiger l'authentification multi-facteur (MFA) afin de protéger les actions d'API sensibles de votre compte.

Pour plus d'informations sur les politiques et les autorisations, AWS consultez les rubriques suivantes :

- Gestion de l'accès aux AWS ressources
- Logique d'évaluation de politiques.
- Managing Access Permissions to Your Amazon S3 Resources (gestion des autorisations d'accès à vos ressources Amazon S3) dans le guide de l'utilisateur du service de stockage simple Amazon.
- Pour savoir si les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, veuillez consulter Qu'est-ce que l'Analyseur d'accès IAM ?.

Balisage des ressources IAM

Une balise est un attribut personnalisé que vous pouvez attribue à une ressource AWS . Chaque balise se compose de deux parties :

- Une clé de balise (par exemple, `CostCenter`, `Environment`, `Project` ou `Purpose`).
- Un champ facultatif appelé valeur de balise (par exemple, `111122223333`, `Production` ou le nom d'une équipe). Omettre la valeur de balise équivaut à l'utilisation d'une chaîne vide.

Ensemble, ces éléments sont connus sous le nom de paires clé-valeur. Pour connaître les limites de nombre de balises possibles pour les ressources IAM, veuillez consulter IAM et quotas AWS STS.

> **Note**
>
> Pour plus d'informations sur la sensibilité à la casse des clés de balises et les valeurs de clés de balises, consultez Case sensitivity (français non garanti).

Les balises vous aident à identifier et à organiser vos AWS ressources. De nombreux AWS services prennent en charge le balisage. Vous pouvez donc attribuer le même tag aux ressources de différents services pour indiquer que les ressources sont liées. Par exemple, vous pouvez attribuer la même balise à un rôle IAM que vous affectez à un compartiment Amazon S3. Pour plus d'informations sur les stratégies de balisage, consultez le Guide de l'utilisateur AWS des ressources de balisage.

Outre l'identification, l'organisation et le suivi de vos ressources IAM avec des balises, vous pouvez utiliser des balises dans les politiques IAM afin de contrôler qui peut consulter et interagir avec vos ressources. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour plus d'informations, consultez Transmission des balises de session AWS STS.

Choisissez une convention de dénomination des AWS balises

Lorsque vous commencez à attacher des balises à vos ressources IAM, choisissez soigneusement votre convention de dénomination de balises. Appliquez la même convention à tous vos AWS tags. Cela est particulièrement important si vous utilisez des balises dans les politiques pour contrôler l'accès aux AWS ressources. Si vous utilisez déjà des balises dans AWS, passez en revue votre convention de dénomination et ajustez-la en conséquence.

> **Note**
>
> Si votre compte est membre de AWS Organizations, consultez les politiques relatives aux tags dans le guide de l'utilisateur des Organizations pour en savoir plus sur l'utilisation des tags dans les Organizations.

Bonnes pratiques pour nommer les balises

Voici quelques bonnes pratiques et conventions pour la dénomination des balises.

Veillez à ce que les noms de balises soient utilisés de manière cohérente. Par exemple, les balises `CostCenter` et `costcenter` sont différentes, de sorte que l'une peut être configurée en tant que balise de répartition des coûts pour l'analyse financière et la confection de rapports, alors que

l'autre peut ne pas l'être. De même, le `Name` tag apparaît dans la AWS console pour de nombreuses ressources, mais pas le `name` tag. Pour plus d'informations sur la sensibilité à la casse des clés de balises et les valeurs de clés de balises, consultez Case sensitivity (français non garanti).

Un certain nombre de balises sont prédéfinies AWS ou créées automatiquement par différents AWS services. De nombreux noms de balises AWS définis utilisent uniquement des minuscules, avec des tirets séparant les mots dans le nom, et des préfixes pour identifier le service source de la balise. Par exemple :

- `aws:ec2spot:fleet-request-id` identifie la demande d'instance Spot Amazon EC2 ponctuelle qui a lancé l'instance.
- `aws:cloudformation:stack-name`identifie la AWS CloudFormation pile qui a créé la ressource.
- `elasticbeanstalk:environment-name` identifie l'application qui a créé la ressource.

Pensez à nommer vos balises en minuscules, avec des traits d'union séparant les mots et un préfixe identifiant le nom de l'organisation ou le nom abrégé. Par exemple, pour une société fictive nommée AnyCompany, vous pouvez définir des balises telles que :

- `anycompany:cost-center` pour identifier le code du centre de coûts interne
- `anycompany:environment-type` pour déterminer si l'environnement est un environnement de développement, de test ou de production
- `anycompany:application-id` pour identifier l'application pour laquelle la ressource a été créée

Le préfixe garantit que les balises sont clairement identifiées comme ayant été définies par votre organisation et AWS non par un outil tiers que vous utilisez peut-être. L'utilisation de minuscules et de traits d'union pour les séparateurs évite toute confusion quant à l'utilisation de majuscules pour le nom d'une balise. Par exemple, `anycompany:project-id`est plus simple à mémoriser que `ANYCOMPANY:ProjectID`, `anycompany:projectID` ou`Anycompany:ProjectId`.

Règles de balisage dans IAM et AWS STS

Un certain nombre de conventions régissent la création et l'application des balises dans IAM et AWS STS.

Dénomination des balises

Respectez les conventions suivantes lors de la formulation d'une convention de dénomination des balises pour les ressources IAM, les sessions d' AWS STS assume-rôle et AWS STS les sessions utilisateur fédérées :

Character requirements (Exigences relatives aux caractères) : les clés et valeurs des balises peuvent inclure n'importe quelle combinaison de lettres, chiffres, espaces et caractères _ . : / = + - @.

Case sensitivity (Sensibilité à la casse) : la sensibilité à la casse pour les clés de balise diffère selon le type de ressource IAM qui est balisé. Les valeurs clés de balise pour les utilisateurs et les rôles IAM ne sont pas sensibles à la casse, mais la casse est conservée. Cela signifie que vous ne pouvez pas avoir des clés de balise **`Department`** et **`department`** distinctes. Si vous avez balisé un utilisateur avec la balise **`Department=finance`** et que vous ajoutez la balise **`department=hr`**, elle remplace la première balise. Une deuxième balise n'est pas ajoutée.

Pour les autres types de ressources IAM, les valeurs de clé de balise sont sensibles à la casse. Cela signifie que vous pouvez avoir des clés de balise **`Costcenter`** et **`costcenter`** séparées. Par exemple, si vous avez balisé une politique gérée par le client avec la balise **`Costcenter = 1234`** et que vous ajoutez la balise **`costcenter = 5678`**, la politique aura à la fois les clés de balise **`Costcenter`** et **`costcenter`**.

À titre de bonne pratique, nous vous recommandons d'éviter d'utiliser des balises similaires avec un traitement de casse incohérent. Nous vous recommandons de choisir une stratégie pour l'utilisation de majuscules pour le nom d'une balise et de mettre en œuvre cette stratégie de manière cohérente sur tous les types de ressources. Pour en savoir plus sur les meilleures pratiques en matière de balisage, consultez la section AWS Ressources relatives au balisage dans le. Références générales AWS

Les listes suivantes présentent les différences de sensibilité à la casse pour les clés de balise attachées aux ressources IAM.

Les valeurs de clé de balise ne sont pas sensibles à la casse :

- Rôles IAM
- Utilisateurs IAM

Les valeurs de clé de balise sont sensibles à la casse :

- Politiques gérées par le client
- Profils d’instance
- Fournisseurs d'identité OpenID Connect
- Fournisseurs d'identité SAML
- Certificats de serveur
- Appareils MFA virtuels

En outre, les règles suivantes s'appliquent :

- Vous ne pouvez pas créer une clé ou valeur de balise qui commence par le texte **`aws:`**. Ce préfixe de balise est réservé à un usage AWS interne.
- Vous pouvez créer une balise avec une valeur vide comme **`phoneNumber =`** . Vous ne pouvez pas créer une clé de balise vide.
- Vous ne pouvez pas spécifier plusieurs valeurs dans une seule balise, mais vous pouvez créer une structure multivaleur personnalisée dans la valeur unique. Par exemple, supposons que l'utilisateur Zhang fonctionne sur l'équipe d'ingénierie et l'équipe d'assurance qualité. Si vous attachez la balise **`team = Engineering`**, puis attachez la balise **`team = QA`**, vous modifiez la valeur de la balise de **`Engineering`** en **`QA`**. Au lieu de cela, vous pouvez inclure plusieurs valeurs dans une seule balise avec un séparateur personnalisé. Dans cet exemple, vous pouvez attacher la balise **`team = Engineering:QA`** à Zhang.

> **Note**
>
> Pour contrôler l'accès aux ingénieurs de cet exemple à l'aide de la balise **`team`**, vous devez créer une politique qui autorise chaque configuration susceptible d'inclure **`Engineering`**, y compris **`Engineering:QA`**. Pour en savoir plus sur l'utilisation des balises dans les politiques consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Application et modification de balises

Observez les conventions suivantes lorsque vous attachez des balises aux ressources IAM :

- Vous pouvez baliser la plupart des ressources IAM, mais pas les groupes, les rôles endossés, les rapports d'accès, les dispositifs MFA matériels.

- Vous ne pouvez pas utiliser Tag Editor pour baliser les ressources IAM. Tag Editor ne prend pas en charge les balises IAM. Pour plus d'informations sur l'utilisation de Tag Editor avec d'autres services, consultez Utilisation de Tag Editor dans le Guide de l'utilisateur AWS Resource Groups .
- Vous devez disposer d'autorisations spécifiques pour baliser une ressource IAM. Pour ajouter ou supprimer des balises aux ressources, vous devez également disposer d'une autorisation pour afficher la liste des balises. Pour plus d'informations, consultez la liste des rubriques pour chaque ressource IAM à la fin de cette page.
- Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d'informations, consultez IAM et quotas AWS STS.
- Vous pouvez appliquer la même balise à plusieurs ressources IAM. Par exemple, supposons que vous ayez un service nommé `AWS_Development` comprenant 12 membres. Vous pouvez avoir 12 utilisateurs et un rôle avec la clé de balise **`department`** et la valeur **`awsDevelopment`** (**`department = awsDevelopment`**). Vous pouvez également utiliser la même balise sur les ressources d'autres services qui prennent en charge le balisage.
- Les entités IAM (utilisateurs ou rôles) ne peuvent pas avoir plusieurs instances de la même clé de balise. Par exemple, si vous avez un utilisateur avec la paire clé-valeur de balise **`costCenter = 1234`**, vous pouvez ensuite attacher la paire clé-valeur de balise **`costCenter = 5678`**. IAM met à jour la valeur de la balise **`costCenter`** à **`5678`**.
- Pour modifier une balise qui est attachée à une entité IAM (utilisateur ou rôle), attachez une balise avec une nouvelle valeur pour remplacer la balise existante. Par exemple, supposons que vous disposiez d'un utilisateur avec la paire clé-valeur de balise **`department = Engineering`**. Si vous devez déplacer l'utilisateur vers le service d'assurance qualité, vous pouvez attacher la paire clé-valeur de balise **`department = QA`** à l'utilisateur. Il s'ensuit que la valeur **`Engineering`** de la clé de balise **`department`** est remplacée par la valeur **`QA`**.

Rubriques

- Étiquette d'utilisateurs IAM
- Étiquette de rôles IAM
- Balisage des politiques gérées par le client
- Balisage de fournisseurs d'identité IAM
- Balisage de profils d'instance pour les rôles Amazon EC2
- Balisage des certificats de serveur
- Balisage des dispositifs MFA virtuels

- Transmission des balises de session AWS STS

Étiquette d'utilisateurs IAM

Vous pouvez utiliser des paires clé-valeur de balise IAM pour ajouter des attributs personnalisés à un utilisateur IAM. Par exemple, pour ajouter les informations relatives à un utilisateur, vous pouvez ajouter la clé de balise **`location`** et la valeur de balise **`us_wa_seattle`**. Ou vous pouvez utiliser trois paires clé-valeur de balise aux emplacements distincts : **`loc-country = us`**, **`loc-state = wa`** et **`loc-city = seattle`**. Vous pouvez utiliser les balises pour contrôler l'accès d'un utilisateur aux ressources d'une entité ou pour contrôler les balises qui peuvent être attachées à un utilisateur. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour plus d'informations, consultez Transmission des balises de session AWS STS.

Autorisations requises pour le balisage des utilisateurs IAM

Vous devez configurer les autorisations pour permettre à un utilisateur IAM de baliser d'autres utilisateurs. Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListUserTags`
- `iam:TagUser`
- `iam:UntagUser`

Pour autoriser un utilisateur IAM à ajouter, afficher la liste ou supprimer une balise pour un utilisateur spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'utilisateur IAM qui doit gérer les balises. Utilisez votre numéro de compte et remplacez *`<username>`* par le nom de l'utilisateur dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
```

```
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Pour autoriser un utilisateur IAM à gérer lui-même les balises

Ajoutez l'instruction suivante à la politique d'autorisations pour les utilisateurs qui autorisent les utilisateurs à gérer leurs propres balises. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::user/${aws:username}"
}
```

Pour autoriser un utilisateur IAM à ajouter une balise à un utilisateur spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'utilisateur IAM qui doit ajouter, mais non pas supprimer, les balises d'un utilisateur spécifique.

> **Note**
>
> L'action `iam:TagUser` nécessite d'inclure également l'action `iam:ListUserTags`.

Pour utiliser cette politique, remplacez *<username>* par le nom de l'utilisateur dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
```

```
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Vous pouvez également utiliser une politique AWS gérée telle que IAM FullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les utilisateurs IAM (console)

Vous pouvez gérer les balises pour les utilisateurs IAM depuis la AWS Management Console.

Pour gérer les balises sur les utilisateurs (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console, choisissez Users (Utilisateurs), puis choisissez le nom de l'utilisateur que vous souhaitez modifier.
3. Choisissez l'onglet Balises, puis effectuez l'une des actions suivantes :
   - Choisissez Ajouter une nouvelle balise si l'utilisateur ne dispose pas encore de balises.
   - Choisissez Manage tags (Gérer les balises) pour gérer l'ensemble de balises existant.
4. Ajoutez ou supprimez des balises pour terminer l'ensemble de balises. Ensuite, choisissez Enregistrer les modifications.

Gestion des balises sur les utilisateurs IAM (AWS CLI ou AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour les utilisateurs IAM. Vous pouvez utiliser l'API AWS CLI ou l' AWS API pour gérer les balises pour les utilisateurs IAM.

Pour répertorier les balises actuellement associées à un utilisateur IAM (AWS CLI ou à une AWS API)

- AWS CLI: cime list-user-tags
- AWS API : ListUserTags

Pour associer des balises à un utilisateur IAM (AWS CLI ou à une AWS API)

- AWS CLI: aws iam tag-user
- AWS API : TagUser

Pour supprimer des balises d'un utilisateur IAM (AWS CLI ou d'une AWS API)

- AWS CLI: aws iam untag-user
- AWS API : UntagUser

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Étiquette de rôles IAM

Vous pouvez utiliser des paires clé-valeur de balises pour ajouter des attributs personnalisés à un rôle IAM. Par exemple, pour ajouter les informations relatives à un rôle, vous pouvez ajouter la clé de balise **`location`** et la valeur de balise **`us_wa_seattle`**. Ou vous pouvez utiliser trois paires clé-valeur de balise aux emplacements distincts : **`loc-country = us`**, **`loc-state = wa`** et **`loc-city = seattle`**. Vous pouvez utiliser les balises pour contrôler l'accès d'un rôle aux ressources d'une entité ou pour contrôler les balises qui peuvent être attachées à un rôle. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour plus d'informations, consultez Transmission des balises de session AWS STS.

Autorisations requises pour le balisage des rôles IAM

Vous devez configurer les autorisations de manière à permettre à un rôle IAM de baliser d'autres entités (utilisateurs ou rôles). Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListRoleTags`

- `iam:TagRole`
- `iam:UntagRole`
- `iam:ListUserTags`
- `iam:TagUser`
- `iam:UntagUser`

Pour autoriser un rôle IAM à ajouter, afficher la liste ou supprimer une balise pour un utilisateur spécifique

Ajoutez l'instruction suivante à la politique d'autorisations du rôle IAM qui doit gérer les balises. Utilisez votre numéro de compte et remplacez *<username>* par le nom de l'utilisateur dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Pour autoriser un rôle IAM à ajouter une balise à un utilisateur spécifique

Ajoutez l'instruction suivante à la politique d'autorisations du rôle IAM qui doit ajouter, mais non pas supprimer, les balises d'un utilisateur spécifique.

Pour utiliser cette politique, remplacez *<username>* par le nom de l'utilisateur dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
```

```
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Pour autoriser un rôle IAM à ajouter, afficher la liste ou supprimer une balise pour un rôle spécifique

Ajoutez l'instruction suivante à la politique d'autorisations du rôle IAM qui doit gérer les balises. Remplacez *<rolename>* par le nom du rôle dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListRoleTags",
        "iam:TagRole",
        "iam:UntagRole"
    ],
    "Resource": "arn:aws:iam::<account-number>:role/<rolename>"
}
```

Vous pouvez également utiliser une politique AWS gérée telle que IAM FullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les rôles IAM (console)

Vous pouvez gérer les balises pour les rôles IAM depuis la AWS Management Console.

Pour gérer les balises sur les rôles (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console, choisissez Roles (Rôles), puis choisissez le nom du rôle que vous souhaitez modifier.
3. Choisissez l'onglet Balises, puis effectuez l'une des actions suivantes :
   - Choisissez Add new tag (Ajouter une nouvelle balise) si le rôle ne dispose pas encore de balises.
   - Choisissez Manage tags (Gérer les balises) pour gérer l'ensemble de balises existant.

4. Ajoutez ou supprimez des balises pour terminer l'ensemble de balises. Ensuite, choisissez Save changes (Enregistrer les modifications).

Gestion des balises sur les rôles IAM (AWS CLI ou AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour les rôles IAM. Vous pouvez utiliser l'API AWS CLI ou l' AWS API pour gérer les balises des rôles IAM.

Pour répertorier les balises actuellement associées à un rôle IAM (AWS CLI ou à une AWS API)

- AWS CLI: cime list-role-tags
- AWS API : ListRoleTags

Pour associer des balises à un rôle IAM (AWS CLI ou à une AWS API)

- AWS CLI: aws iam tag-role
- AWS API : TagRole

Pour supprimer des balises d'un rôle IAM (AWS CLI ou d'une AWS API)

- AWS CLI: aws iam untag-role
- AWS API : UntagRole

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Balisage des politiques gérées par le client

Vous pouvez utiliser des paires clé-valeur de balises IAM pour ajouter des attributs personnalisés à vos politiques gérées par le client. Par exemple, pour baliser une politique avec des informations de service, vous pouvez ajouter la clé de balise **`Department`** et la valeur de balise **`eng`**. Ou bien, vous pouvez baliser les politiques pour indiquer qu'elles sont destinées à un environnement spécifique, comme **`Environment = lab`**. Vous pouvez utiliser les balises pour contrôler l'accès aux ressources ou pour contrôler les balises qui peuvent être attachées à une ressource. Pour en savoir plus

sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour plus d'informations, consultez Transmission des balises de session AWS STS.

Autorisations requises pour baliser les politiques gérées par le client

Vous devez configurer des autorisations pour autoriser une entité IAM (utilisateurs ou rôles) à baliser les politiques gérées par le client. Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListPolicyTags`
- `iam:TagPolicy`
- `iam:UntagPolicy`

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter, afficher la liste ou supprimer une balise pour une politique gérée par le client

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit gérer les balises. Utilisez votre numéro de compte et remplacez *<policyname>* par le nom de la politique dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called "Création de politiques à l'aide de l'éditeur JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy",
        "iam:UntagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter une balise à une politique gérée par le client spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit ajouter, mais non pas supprimer, les balises d'une politique spécifique.

> **Note**
>
> L'action `iam:TagPolicy` nécessite d'inclure également l'action `iam:ListPolicyTags`.

Pour utiliser cette politique, remplacez *<policyname>* par le nom de la politique dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

Vous pouvez également utiliser une politique AWS gérée telle que IAM FullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les politiques gérées par le client IAM (console)

Vous pouvez gérer les balises pour les politiques gérées par le client IAM à partir de la AWS Management Console.

Pour gérer les balises sur les politiques gérées par le client (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console, choisissez Policies (Politiques), puis choisissez le nom de la politique gérée par le client que vous souhaitez modifier.
3. Choisissez l'onglet Balises, puis Gérer les balises.
4. Ajoutez ou supprimez des balises pour terminer l'ensemble de balises. Ensuite, choisissez Enregistrer les modifications.

Gestion des balises sur les politiques (AWS CLI ou AWS API) gérées par le client IAM

Vous pouvez afficher la liste, attacher ou supprimer des balises pour les politiques gérées par le client IAM. Vous pouvez utiliser l'API AWS CLI ou l' AWS API pour gérer les balises dans le cadre des politiques gérées par les clients IAM.

Pour répertorier les balises actuellement attachées à une politique (AWS CLI ou AWS API) gérée par le client IAM

- AWS CLI: vison list-policy-tags
- AWS API : ListPolicyTags

Pour associer des balises à une politique (AWS CLI ou AWS API) gérée par le client IAM

- AWS CLI : aws iam tag-policy
- AWS API : TagPolicy

Pour supprimer des balises d'une politique (AWS CLI ou AWS API) gérée par le client IAM

- AWS CLI : aws iam untag-policy
- AWS API : UntagPolicy

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Balisage de fournisseurs d'identité IAM

Vous pouvez utiliser des paires clé-valeur de balise IAM pour ajouter des attributs personnalisés aux fournisseurs d'identité IAM (). IdPs

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour plus d’informations, consultez Transmission des balises de session AWS STS.

Pour en savoir plus sur le balisage IdPs dans IAM, consultez les rubriques suivantes :

Rubriques

- Balisage de fournisseurs d'identité OpenID Connect (OIDC)
- Balisage de fournisseurs d'identité SAML IAM

Balisage de fournisseurs d'identité OpenID Connect (OIDC)

Vous pouvez utiliser des paires clé-valeur de balises IAM pour ajouter des attributs personnalisés aux fournisseurs d'identité OpenID Connect (OIDC) IAM. Par exemple, pour identifier un fournisseur d'identité OIDC, vous pouvez ajouter la clé de balise **`google`** et la valeur de balise **`oidc`**. Vous pouvez utiliser les balises pour contrôler l'accès aux ressources ou pour contrôler les balises qui peuvent être attachées à un objet. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Autorisations requises pour baliser les fournisseurs d'identité OIDC IAM

Vous devez configurer les autorisations de manière à permettre à une entité IAM (utilisateur ou rôle) de baliser des fournisseurs d'identité OIDC IAM. Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListOpenIDConnectProviderTags`
- `iam:TagOpenIDConnectProvider`
- `iam:UntagOpenIDConnectProvider`

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter, afficher la liste ou supprimer une balise pour un fournisseur d'identité OIDC IAM

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit gérer les balises. Utilisez votre numéro de compte et remplacez *`<OIDC ProviderName >`* par le nom du fournisseur OIDC dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider",
```

```
        "iam:UntagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter une balise à un fournisseur d'identité OIDC IAM spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit ajouter, mais non pas supprimer, les balises d'un fournisseur d'identité spécifique.

> **Note**
>
> L'action `iam:TagOpenIDConnectProvider` nécessite d'inclure également l'action `iam:ListOpenIDConnectProviderTags`.

Pour utiliser cette politique, remplacez *<OIDC ProviderName >* par le nom du fournisseur OIDC dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

Vous pouvez également utiliser une politique AWS gérée telle que IAM FullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les fournisseurs d'identité OIDC IAM (console)

Vous pouvez gérer les balises pour les fournisseurs d'identité OIDC IAM à partir de la AWS Management Console.

Pour gérer les balises sur les fournisseurs d'identité OIDC (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console, sélectionnez Identity providers (Fournisseurs d'identité), puis le nom du fournisseur d'identité que vous souhaitez modifier.
3. Dans la section Tags (Balises), choisissez Manage Tags (Gérer les balises), puis effectuez l'une des actions suivantes :
   - Choisissez Add tag (Ajouter une balise) si le fournisseur d'identité OIDC n'a pas encore de balises ou pour ajouter une nouvelle balise.
   - Modifiez les clés et les valeurs de balise existantes.
   - Choisissez Remove tag (Supprimer une balise) pour supprimer une balise.
4. Ensuite, choisissez Enregistrer les modifications.

Gestion des balises sur les fournisseurs d'identité IAM OIDC (AWS CLI ou AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour les fournisseurs d'identité OIDC IAM. Vous pouvez utiliser l'API AWS CLI ou l' AWS API pour gérer les balises des fournisseurs d'identité IAM OIDC.

Pour répertorier les balises actuellement attachées à un fournisseur d'identité (AWS CLI ou AWS API) IAM OIDC

- AWS CLI: balises aws iam -provider-tags list-open-id-connect
- AWS API : ListOpenID ConnectProviderTags

Pour associer des balises à un fournisseur d'identité (AWS CLI ou AWS API) IAM OIDC

- AWS CLI: fournisseur AWS iam tag-open-id-connect
- AWS API : TagOpenID ConnectProvider

Pour supprimer des balises d'un fournisseur d'identité (AWS CLI ou AWS API) IAM OIDC

- AWS CLI: fournisseur AWS iam untag-open-id-connect
- AWS API : UntagOpenID ConnectProvider

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Balisage de fournisseurs d'identité SAML IAM

Vous pouvez utiliser des paires clé-valeur de balises IAM pour ajouter des attributs personnalisés aux fournisseurs d'identité SAML. Par exemple, pour identifier un fournisseur, vous pouvez ajouter la clé de balise **`okta`** et la valeur de balise **`saml`**. Vous pouvez utiliser les balises pour contrôler l'accès aux ressources ou pour contrôler les balises qui peuvent être attachées à un objet. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Autorisations requises pour baliser les fournisseurs d'identité SAML

Vous devez configurer les autorisations pour permettre à une entité IAM (utilisateurs ou rôles) de baliser les fournisseurs d'identité basés sur SAML 2.0 (). IdPs Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListSAMLProviderTags`
- `iam:TagSAMLProvider`
- `iam:UntagSAMLProvider`

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter, afficher la liste ou supprimer une balise pour un fournisseur d'identité SAML

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit gérer les balises. Utilisez votre numéro de compte et remplacez *`<SAML ProviderName >`* par le nom du fournisseur SAML dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
```

```
        "iam:TagSAMLProvider",
        "iam:UntagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter une balise à un fournisseur d'identité SAML spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit ajouter, mais non pas supprimer, les balises d'un fournisseur d'identité SAML spécifique.

> **Note**
>
> L'action `iam:TagSAMLProvider` nécessite d'inclure également l'action `iam:ListSAMLProviderTags`.

Pour utiliser cette politique, remplacez *<SAML ProviderName >* par le nom du fournisseur SAML dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
        "iam:TagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

Vous pouvez également utiliser une politique AWS gérée telle que IAM FullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les fournisseurs d'identité SAML IAM (console)

Vous pouvez gérer les balises pour les fournisseurs d'identité SAML IAM à partir de la AWS Management Console.

Pour gérer les balises sur les fournisseurs d'identité SAML (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console, sélectionnez Identity providers (Fournisseurs d'identité), puis le nom du fournisseur d'identité SAML que vous souhaitez modifier.
3. Dans la section Tags (Balises), choisissez Manage Tags (Gérer les balises), puis effectuez l'une des actions suivantes :
   - Choisissez Add tag (Ajouter une balise) si le fournisseur d'identité SAML n'a pas encore de balises ou pour ajouter une nouvelle balise.
   - Modifiez les clés et les valeurs de balise existantes.
   - Choisissez Remove tag (Supprimer une balise) pour supprimer une balise.
4. Ajoutez ou supprimez des balises pour terminer l'ensemble de balises. Ensuite, choisissez Enregistrer les modifications.

Gestion des balises sur les fournisseurs d'identité IAM SAML (AWS CLI ou AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour les fournisseurs d'identité SAML IAM. Vous pouvez utiliser l'API AWS CLI ou l' AWS API pour gérer les balises des fournisseurs d'identité IAM SAML.

Pour répertorier les balises actuellement attachées à un fournisseur d'identité SAML (AWS CLI ou AWS API)

- AWS CLI: cime list-saml-provider-tags
- AWS API : ListSAML ProviderTags

Pour associer des balises à un fournisseur d'identité (AWS CLI ou AWS API) SAML

- AWS CLI: cime tag-saml-provider
- AWS API : TagSamlProvider

Pour supprimer des balises d'un fournisseur d'identité (AWS CLI ou d'une AWS API) SAML

- AWS CLI: cime untag-saml-provider

- AWS API : UntagSamlProvider

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Balisage de profils d'instance pour les rôles Amazon EC2

Lorsque vous lancez une instance Amazon EC2, vous spécifiez un rôle IAM à associer à celle-ci. Un profil d'instance est un conteneur pour un rôle IAM que vous pouvez utiliser pour transmettre les informations liées au rôle à une instance Amazon EC2 lorsque l'instance démarre. Vous pouvez baliser les profils d'instance lorsque vous utilisez l' AWS API AWS CLI or.

Vous pouvez utiliser des paires clé-valeur de balise IAM pour ajouter des attributs personnalisés à un profil d'instance. Par exemple, pour ajouter des informations de service à un profil d'instance, vous pouvez ajouter la clé de balise **`access-team`** et la valeur de balise **`eng`**. Cela donne aux principaux avec des balises correspondantes l'accès aux profils d'instance avec la même balise. Vous pouvez utiliser plusieurs paires clé-valeur de balise pour spécifier une équipe et un projet : **`access-team = eng`** , et **`project = peg`**. Vous pouvez utiliser les balises pour contrôler l'accès d'un utilisateur aux ressources d'une entité ou pour contrôler les balises qui peuvent être attachées à un utilisateur. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour plus d'informations, consultez Transmission des balises de session AWS STS.

Autorisations requises pour le balisage des profils d'instance

Vous devez configurer les autorisations de manière à permettre à une entité IAM (utilisateur ou rôle) de baliser des profils d'instance. Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListInstanceProfileTags`
- `iam:TagInstanceProfile`
- `iam:UntagInstanceProfile`

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter, afficher la liste ou supprimer une balise pour un profil d'instance

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit gérer les balises. Utilisez votre numéro de compte et remplacez *< InstanceProfileName >* par le nom du profil d'instance dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
        "iam:TagInstanceProfile",
        "iam:UntagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter une balise à un profil d'instance spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit ajouter, mais non pas supprimer, les balises d'un profil d'instance spécifique.

> **Note**
>
> L'action `iam:TagInstanceProfile` nécessite d'inclure également l'action `iam:ListInstanceProfileTags`.

Pour utiliser cette politique, remplacez *< InstanceProfileName >* par le nom du profil d'instance dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
        "iam:TagInstanceProfile"
    ],
```

```
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

Vous pouvez également utiliser une politique AWS gérée telle que IAM FullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les profils d'instance (AWS CLI ou AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour des profils d'instance. Vous pouvez utiliser l'API AWS CLI ou l' AWS API pour gérer les balises des profils d'instance.

Pour répertorier les balises actuellement attachées à un profil d'instance (AWS CLI ou à une AWS API)

- AWS CLI: cime list-instance-profile-tags
- AWS API : ListInstanceProfileTags

Pour associer des balises à un profil d'instance (AWS CLI ou à une AWS API)

- AWS CLI: cime tag-instance-profile
- AWS API : TagInstanceProfile

Pour supprimer des balises d'un profil d'instance (AWS CLI ou d'une AWS API)

- AWS CLI: cime untag-instance-profile
- AWS API : UntagInstanceProfile

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Balisage des certificats de serveur

Si vous utilisez IAM pour gérer les certificats SSL/TLS, vous pouvez baliser les certificats de serveur dans IAM à l'aide de l'API or. AWS CLI AWS Pour les certificats d'une région prise en charge par AWS Certificate Manager (ACM), nous vous recommandons d'utiliser ACM au lieu d'IAM pour

provisionner, gérer et déployer vos certificats de serveur. Dans les régions non prises en charge, vous devez utiliser IAM en tant que gestionnaire de certificats. Pour savoir quelles régions sont prises en charge par ACM, consultez Points de terminaison et quotas AWS Certificate Manager (français non garanti) dans Références générales AWS.

Vous pouvez utiliser des paires clé-valeur de balise IAM pour ajouter des attributs personnalisés à un certificat de serveur. Par exemple, pour ajouter des informations sur le propriétaire ou l'administrateur d'un certificat de serveur, ajoutez la clé de balise **`owner`** et la valeur de balise **`net-eng`**. Vous pouvez également spécifier un centre de coûts en ajoutant la clé de balise **`CostCenter`** et la valeur de balise **`1234`**. Vous pouvez utiliser les balises pour contrôler l'accès aux ressources ou pour contrôler les balises qui peuvent être attachées à des ressources. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour plus d'informations, consultez Transmission des balises de session AWS STS.

Autorisations requises pour le balisage des certificats de serveur

Vous devez configurer les autorisations de manière à permettre à une entité IAM (utilisateur ou rôle) de baliser des certificats de serveur. Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListServerCertificateTags`
- `iam:TagServerCertificate`
- `iam:UntagServerCertificate`

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter, afficher la liste ou supprimer une balise pour un certificat de serveur

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit gérer les balises. Utilisez votre numéro de compte et remplacez *`< CertificateName >`* par le nom du certificat de serveur dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
```

```
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate",
        "iam:UntagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
}
```

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter une balise à un certificat de serveur spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit ajouter, mais non pas supprimer, les balises d'un certificat de serveur spécifique.

> **Note**
>
> L'action `iam:TagServerCertificate` nécessite d'inclure également l'action `iam:ListServerCertificateTags`.

Pour utiliser cette politique, remplacez *< CertificateName >* par le nom du certificat de serveur dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
}
```

Vous pouvez également utiliser une politique AWS gérée telle que IAM FullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les certificats de serveur (AWS CLI ou AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour les certificats de serveur. Vous pouvez utiliser l'API AWS CLI ou l' AWS API pour gérer les balises des certificats de serveur.

Pour répertorier les balises actuellement attachées à un certificat de serveur (AWS CLI ou à une AWS API)

- AWS CLI: cime list-server-certificate-tags
- AWS API : ListServerCertificateTags

Pour associer des balises à un certificat de serveur (AWS CLI ou à une AWS API)

- AWS CLI: cime tag-server-certificate
- AWS API : TagServerCertificate

Pour supprimer des balises d'un certificat de serveur (AWS CLI ou d'une AWS API)

- AWS CLI: cime untag-server-certificate
- AWS API : UntagServerCertificate

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Balisage des dispositifs MFA virtuels

Vous pouvez utiliser des paires clé-valeur de balise IAM pour ajouter des attributs personnalisés à un appareil MFA virtuel. Par exemple, pour ajouter des informations de centre de coûts pour l'appareil MFA virtuel d'un utilisateur, vous pouvez ajouter la clé de balise **`CostCenter`** et la valeur de balise **`1234`**. Vous pouvez utiliser les balises pour contrôler l'accès aux ressources ou pour contrôler les balises qui peuvent être attachées à un objet. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Vous pouvez également utiliser des balises AWS STS pour ajouter des attributs personnalisés lorsque vous assumez un rôle ou que vous fédérez un utilisateur. Pour plus d’informations, consultez Transmission des balises de session AWS STS.

Autorisations requises pour le balisage des appareils MFA virtuels

Vous devez configurer les autorisations de manière à permettre à une entité IAM (utilisateur ou rôle) de baliser des appareils MFA virtuels. Vous pouvez spécifier une ou toutes les actions suivantes d'une balise IAM dans une politique IAM :

- `iam:ListMFADeviceTags`
- `iam:TagMFADevice`
- `iam:UntagMFADevice`

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter, afficher la liste ou supprimer une balise pour un appareil MFA virtuel

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit gérer les balises. Utilisez votre numéro de compte et remplacez *<MFATokenID>* par le nom de l'dispositif MFA virtuel dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice",
        "iam:UntagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

Pour autoriser une entité IAM (utilisateur ou rôle) à ajouter une balise à un dispositif MFA virtuel spécifique

Ajoutez l'instruction suivante à la politique d'autorisations de l'entité IAM qui doit ajouter, mais non pas supprimer, les balises d'un appareil MFA spécifique.

> **Note**
>
> L'action `iam:TagMFADevice` nécessite d'inclure également l'action `iam:ListMFADeviceTags`.

Pour utiliser cette politique, remplacez *<MFATokenID>* par le nom de l'dispositif MFA virtuel dont les balises doivent être gérées. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

Vous pouvez également utiliser une politique AWS gérée telle que IAM FullAccess pour fournir un accès complet à IAM.

Gestion des balises sur les appareils MFA virtuels (AWS CLI ou AWS API)

Vous pouvez afficher la liste, attacher ou supprimer des balises pour un appareil MFA virtuel. Vous pouvez utiliser l'API AWS CLI ou l' AWS API pour gérer les balises d'un appareil MFA virtuel.

Pour répertorier les balises actuellement attachées à un périphérique MFA virtuel (AWS CLI ou AWS API)

- AWS CLI: cime list-mfa-device-tags
- AWS API : ListMFA DeviceTags

Pour associer des balises à un périphérique MFA virtuel (AWS CLI ou AWS API)

- AWS CLI: cime tag-mfa-device
- AWS API : TagMFAdevice

Pour supprimer des balises d'un périphérique MFA virtuel (AWS CLI ou AWS d'une API)

- AWS CLI: cime untag-mfa-device
- AWS API : UntagMFAdevice

Pour plus d'informations sur l'attachement de balises aux ressources d'autres AWS services, consultez la documentation de ces services.

Pour plus d'informations sur l'utilisation des balises pour définir d'autres autorisations détaillées avec des politiques d'autorisations IAM, consultez Éléments des politiques IAM : variables et balises.

Transmission des balises de session AWS STS

Les balises de session sont des attributs de paire clé-valeur que vous transmettez lorsque vous endossez un rôle IAM ou fédérez un utilisateur dans AWS STS. Pour ce faire, vous devez effectuer une AWS CLI demande d' AWS API via AWS STS ou via votre fournisseur d'identité (IdP). Lorsque vous demandez des informations AWS STS d'identification de sécurité temporaires, vous générez une session. Les sessions expirent et disposent d'informations d'identification, telles qu'une paire de clés d'accès et un jeton de session. Lorsque vous utilisez les informations d'identification de session pour effectuer une demande ultérieure, le contexte de demande inclut la clé de contexte `aws:PrincipalTag`. Vous pouvez utiliser la clé `aws:PrincipalTag` dans l'élément `Condition` de vos politiques pour autoriser ou refuser l'accès en fonction de ces balises.

Lorsque vous utilisez des informations d'identification temporaires pour effectuer une demande, votre principal peut inclure un ensemble d'étiquettes. Ces balises proviennent des sources suivantes :

1. Balises de session : balises transmises lorsque vous assumez le rôle ou que vous fédérez l'utilisateur à l'aide de l' AWS API AWS CLI or. Pour plus d'informations sur ces opérations, consultez Opérations de balisage de session.
2. Incoming transitive session tags (Balises de session transitive entrantes) : ces balises ont été héritées d'une session précédente dans une chaîne de rôles. Pour plus d'informations, consultez Chaînage des rôles avec des balises de session plus loin dans cette rubrique.
3. Balises IAM : balises attachées à votre rôle IAM endossé.

Rubriques

- Opérations de balisage de session
- Choses à savoir sur les balises de session
- Autorisations requises pour ajouter des balises de session
- Transmission de balises de session en utilisant AssumeRole
- Transmission de balises de session à l'aide de AssumeRoleWith SAML
- Transmission de balises de session en utilisant AssumeRoleWithWebIdentity

- Transmission de balises de session en utilisant GetFederationToken
- Chaînage des rôles avec des balises de session
- Utilisation des balises de session pour ABAC
- Afficher les balises de session dans CloudTrail

Opérations de balisage de session

Vous pouvez transmettre des balises de session à l'aide des opérations suivantes AWS CLI ou des opérations d' AWS API AWS STS. La fonctionnalité AWS Management Console Switch Role ne vous permet pas de transmettre des balises de session.

Vous pouvez également définir les balises de session comme transitives. Les balises transitives persistent pendant le chaînage des rôles. Pour plus d'informations, veuillez consulter Chaînage des rôles avec des balises de session.

Comparaison des méthodes de transmission des balises de session

Opération	Qui peut endosser le rôle	Méthode pour transmettre des balises	Méthode pour définir des balises transitives
Opération d'interface de ligne de commande (CLI) `assume-role` ou d'API `AssumeRole`	Utilisateur ou session IAM	Paramètre API `Tags` ou option d'interface de ligne de commande (CLI) `--tags`	Paramètre API `TransitiveTagKeys` ou option d'interface de ligne de commande (CLI) `--transitive-tag-keys`
Opération d'interface de ligne de commande (CLI) `assume-role-with-saml` ou API `AssumeRoleWithSAML`	Tout utilisateur authentifié à l'aide d'un fournisseur d'identité SAML	Attribut SAML `PrincipalTag`	Attribut SAML `TransitiveTagKeys`

Opération	Qui peut endosser le rôle	Méthode pour transmettre des balises	Méthode pour définir des balises transitives
Opération d'interface de ligne de commande (CLI) `assume-role-with-web-identity` ou API `AssumeRoleWithWebIdentity`	Tout utilisateur authentifié à l'aide d'un fournisseur OIDC	`Principal Tag` jeton OIDC	`TransitiveTagKeys` jeton OIDC
Opération d'interface de ligne de commande (CLI) `get-federation-token` ou API `GetFederationToken`	Utilisateur IAM ou utilisateur racine	Paramètre API `Tags` ou option d'interface de ligne de commande (CLI) `--tags`	Non pris en charge

Les opérations qui prennent en charge le balisage de session peuvent échouer si l'une des conditions suivantes est remplie :

- Vous transmettez plus de 50 balises de session.
- Le texte brut de vos clés de balise de session dépasse 128 caractères.
- Le texte brut des valeurs de vos balises de session dépasse 256 caractères.
- La taille totale du texte brut des politiques de session dépasse 2 048 caractères.
- La taille totale des politiques et balises de session combinées est trop grande. Si l'opération échoue, le message d'erreur indique à quel point les politiques et les balises combinées se rapprochent de la limite de taille supérieure, en pourcentage.

Choses à savoir sur les balises de session

Avant d'utiliser des balises de session, veuillez consulter les détails suivants concernant les sessions et les balises.

- Lorsque vous utilisez des balises de session, les politiques d'approbation pour tous les rôles connectés au fournisseur d'identité qui transmet des balises doivent disposer de l'autorisation `sts:TagSession`. Pour les rôles qui ne disposent pas de cette autorisation dans la politique d'approbation, l'opération `AssumeRole` échoue.
- Lorsque vous demandez une session, vous pouvez spécifier des balises de principal comme balises de session. Les balises s'appliquent aux demandes que vous effectuez à l'aide des informations d'identification de la session.
- Les balises utilisent des paires clé-valeur. Par exemple, pour ajouter des informations de contact à une session, vous pouvez ajouter la clé de balise de session `email` et la valeur de balise `johndoe@example.com`.
- Les balises de session doivent suivre les règles de dénomination des balises dans IAM et AWS STS. Cette rubrique contient des informations sur la sensibilité à la casse et les préfixes restreints qui s'appliquent à vos balises de session.
- Les nouvelles balises de session remplacent les balises de rôle ou d'utilisateur fédéré existantes avec la même clé de balise, quel que soit le cas de caractères.
- Vous ne pouvez pas transmettre de balises de session à l'aide du AWS Management Console.
- Les balises de session ne sont valides que pour la session en cours.
- Les balises de session prennent en charge le chaînage des rôles. Par défaut, AWS STS ne transmet pas de balises aux sessions de rôle suivantes. Toutefois, vous pouvez définir les balises de session comme transitives. Les balises transitives persistent durant le chaînage des rôles et remplacent la mise en correspondance de valeurs `ResourceTag` après l'évaluation de la politique de confiance de rôle. Pour plus d'informations, consultez Chaînage des rôles avec des balises de session.
- Vous pouvez utiliser des balises de session pour contrôler l'accès aux ressources ou pour contrôler les balises qui peuvent être transmises à une session ultérieure. Pour plus d'informations, veuillez consulter Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC.
- Vous pouvez afficher les balises principales de votre session, y compris les balises de session, dans les journaux AWS CloudTrail . Pour plus d'informations, consultez Afficher les balises de session dans CloudTrail.

- Vous devez transmettre une valeur unique pour chaque balise de session. AWS STS ne prend pas en charge les balises de session à valeurs multiples.
- Vous pouvez transmettre un maximum de 50 balises de session. Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d'informations, consultez IAM et quotas AWS STS.
- Une AWS conversion compresse les politiques de session et les balises de session adoptées combinées dans un format binaire compressé avec une limite distincte. Si vous dépassez cette limite, le message d'erreur de l' AWS API AWS CLI or indique dans quelle mesure les politiques et les balises combinées se rapprochent de la limite de taille supérieure, en pourcentage.

Autorisations requises pour ajouter des balises de session

En plus de l'action qui correspond à l'opération d'API, vous devez disposer de l'action d'autorisation suivante dans votre politique :

```
sts:TagSession
```

> **Important**
>
> Lorsque vous utilisez des balises de session, les politiques d'approbation de rôle pour tous les rôles connectés à un fournisseur d'identité doivent disposer de l'autorisation `sts:TagSession`. L'opération `AssumeRole` échoue pour tout rôle connecté à un fournisseur d'identité qui transmet des balises de session sans cette autorisation. Si vous ne souhaitez pas mettre à jour la politique d'approbation de rôle pour chaque rôle, vous pouvez utiliser une instance de fournisseur d'identité distincte pour transmettre les balises de session. Ajoutez ensuite l'autorisation `sts:TagSession` uniquement aux rôles qui sont connectés au fournisseur d'identité distinct.

Vous pouvez utiliser l'action `sts:TagSession` avec les clés de condition suivantes.

- `aws:PrincipalTag` : compare la balise attachée au principal effectuant la demande avec la balise spécifiée dans la politique. Par exemple, vous pouvez autoriser un principal à transmettre des balises de session uniquement si le principal qui effectue la demande possède les balises spécifiées.
- `aws:RequestTag` : compare la paire clé-valeur de balise qui a été transmise dans la demande avec la paire de balises que vous indiquez dans la politique. Par exemple, vous pouvez autoriser

le principal à transmettre les balises de session spécifiées, mais uniquement avec les valeurs spécifiées.

- `aws:ResourceTag` : compare la paire clé-valeur de balise que vous avez spécifiée dans la politique avec la paire clé-valeur attachée à la ressource. Par exemple, vous pouvez autoriser le principal à transmettre des balises de session uniquement si le rôle qu'il endosse inclut les balises spécifiées.
- `aws:TagKeys` : compare les clés de balise d'une demande avec les clés que vous avez spécifiées dans la politique. Par exemple, vous pouvez autoriser le principal à transmettre uniquement les balises de session avec les clés de balise spécifiées. Cette clé de condition limite l'ensemble maximal de balises de session pouvant être passé.
- `sts:TransitiveTagKeys` : compare les clés des balises de session transitive de la demande à celles spécifiées dans la politique. Par exemple, vous pouvez écrire une politique pour permettre à un principal de définir uniquement des balises spécifiques comme transitives. Les balises transitives persistent pendant le chaînage des rôles. Pour plus d'informations, veuillez consulter Chaînage des rôles avec des balises de session.

Par exemple, la politique d'approbation de rôle suivante permet à l'utilisateur `test-session-tags` d'endosser le rôle auquel la politique est attachée. Lorsque cet utilisateur assume le rôle, il doit utiliser l' AWS API AWS CLI or pour transmettre les trois balises de session requises et l'ID externe requis. En outre, l'utilisateur peut choisir de définir les balises `Department` et `Project` comme transitives.

Example Politique d'approbation de rôle pour les balises de session

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIamUserAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-tags"},
            "Condition": {
              "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*",
                    "aws:RequestTag/Department": "*"
                },
                "StringEquals": {"sts:ExternalId": "Example987"}
```

```
            }
        },
        {
            "Sid": "AllowPassSessionTagsAndTransitive",
            "Effect": "Allow",
            "Action": "sts:TagSession",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-tags"},
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*"
                },
                "StringEquals": {
                    "aws:RequestTag/Department": [
                        "Engineering",
                        "Marketing"
                    ]
                },
                "ForAllValues:StringEquals": {
                    "sts:TransitiveTagKeys": [
                        "Project",
                        "Department"
                    ]
                }
            }
        }
    ]
}
```

À quoi sert cette politique ?

- La instruction `AllowIamUserAssumeRole` permet à l'utilisateur `test-session-tags` d'endosser le rôle auquel la politique est attachée. Lorsque cet utilisateur endosse le rôle, il doit passer les balises de session et l'ID externe requis.
  - Le premier bloc de condition de cette instruction nécessite que l'utilisateur transmette les balises de session `Project`, `CostCenter` et `Department`. Les valeurs de balise n'ont pas d'importance dans cette instruction, vous pouvez donc utiliser des caractères génériques (*) pour les valeurs de balise. Ce bloc garantit que l'utilisateur transmette au moins ces trois balises de session. Sinon, l'opération échoue. L'utilisateur peut transmettre des balises supplémentaires.
  - Le deuxième bloc de condition nécessite que l'utilisateur transmette un ID externe avec la valeur `Example987`.

- L'instruction `AllowPassSessionTagsAndTransitive` autorise l'action d'autorisations `sts:TagSession`. Cette action doit être autorisée avant que l'utilisateur puisse transmettre des balises de session. Si votre politique inclut la première instruction sans la deuxième, l'utilisateur ne peut pas endosser le rôle.
  - Le premier bloc de condition de cette instruction permet à l'utilisateur de transmettre n'importe quelle valeur pour les balises de session `CostCenter` et `Project`. Pour ce faire, utilisez des caractères génériques (*) pour la valeur de balise dans la politique, ce qui nécessite l'utilisation de l'opérateur de StringLikecondition.
  - Le deuxième bloc de condition permet à l'utilisateur de transmettre uniquement la valeur `Engineering` ou `Marketing` de la balise de session `Department`.
  - Le troisième bloc de condition répertorie l'ensemble maximal de balises que vous pouvez définir comme transitives. L'utilisateur peut choisir de définir un sous-ensemble ou aucune balise comme transitive. Il ne peut pas définir de balises supplémentaires comme transitives. Vous pouvez exiger qu'il définisse au moins une des balises comme transitive en ajoutant un autre bloc de condition qui inclut `"Null":{"sts:TransitiveTagKeys":"false"}`.

Transmission de balises de session en utilisant AssumeRole

L'`AssumeRole`opération renvoie un ensemble d'informations d'identification temporaires que vous pouvez utiliser pour accéder aux AWS ressources. Vous pouvez utiliser l'utilisateur ou les informations d'identification du rôle IAM pour appeler `AssumeRole`. Pour transmettre des balises de session tout en assumant un rôle, utilisez l'`--tags` AWS CLI option ou le paramètre `Tags` AWS API.

Pour définir les balises comme transitives, utilisez l'`--transitive-tag-keys` AWS CLI option ou le paramètre `TransitiveTagKeys` AWS API. Les balises transitives persistent pendant le chaînage des rôles. Pour plus d'informations, veuillez consulter Chaînage des rôles avec des balises de session.

L'exemple suivant montre un exemple de demande qui utilise `AssumeRole`. Dans cet exemple, lorsque vous endossez le rôle `my-role-example`, vous créez une session nommée `my-session`. Vous ajoutez les paires clé-valeur de balise de session `Project` = `Automation`, `CostCenter` = `12345` et `Department` = `Engineering`. Vous définissez également les balises `Project` et `Department` comme transitives en spécifiant leurs clés.

Example Exemple de demande AssumeRole CLI

```
aws sts assume-role \
```

```
--role-arn arn:aws:iam::123456789012:role/my-role-example \
--role-session-name my-session \
--tags Key=Project,Value=Automation Key=CostCenter,Value=12345
 Key=Department,Value=Engineering \
--transitive-tag-keys Project Department \
--external-id Example987
```

Transmission de balises de session à l'aide de AssumeRoleWith SAML

L'opération `AssumeRoleWithSAML` authentifie à l'aide de la fédération basée sur SAML. Cette opération renvoie un ensemble d'informations d'identification temporaires que vous pouvez utiliser pour accéder aux AWS ressources. Pour plus d'informations sur l'utilisation de la fédération basée sur SAML pour l' AWS Management Console accès, consultez. Permettre aux utilisateurs fédérés SAML 2.0 d'accéder au AWS Management Console Pour plus de détails sur AWS CLI AWS l'accès à l'API, consultezFédération SAML 2.0. Pour un didacticiel sur la configuration de la fédération SAML pour vos utilisateurs d'Active Directory, consultez la section Authentification AWS fédérée avec les services de fédération Active Directory (ADFS) dans le AWS blog de sécurité.

En tant qu'administrateur, vous pouvez autoriser les membres du répertoire de votre entreprise à se fédérer pour AWS utiliser cette AWS STS `AssumeRoleWithSAML` opération. Pour ce faire, effectuez les tâches suivantes :

1. Configurer votre réseau en tant que fournisseur SAML pour l'interface AWS
2. Créer un fournisseur SAML dans IAM
3. Configurer un rôle et des autorisations dans AWS pour vos utilisateurs fédérés
4. Achever la configuration de l'IdP SAML et créer des assertions pour la réponse d'authentification SAML

AWS inclut des fournisseurs d'identité dotés d'une end-to-end expérience certifiée en matière de balises de session dans le cadre de leurs solutions d'identité. Pour savoir comment utiliser ces fournisseurs d'identité pour configurer des balises de session, veuillez consulter Intégrez des fournisseurs de solutions SAML tiers avec AWS.

Pour transmettre les attributs SAML en tant que balises de session, incluez l'élément `Attribute` dont l'attribut `Name` est défini sur `https://aws.amazon.com/SAML/Attributes/PrincipalTag:`*`{TagKey}`*. Utilisez l'élément `AttributeValue` pour spécifier la valeur de la balise. Incluez un élément `Attribute` distinct pour chaque balise de session.

Par exemple, supposons que vous souhaitez transmettre les attributs d'identité suivants en tant que balises de session :

- `Project:Automation`
- `CostCenter:12345`
- `Department:Engineering`

Pour transmettre ces attributs, incluez les éléments suivants dans votre assertion SAML.

Example Extrait d'une assertion SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Automation</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Department">
  <AttributeValue>Engineering</AttributeValue>
</Attribute>
```

Pour définir les balises précédentes ci-dessus comme transitives, incluez un autre élément `Attribute` dont l'attribut `Name` est défini sur `https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys`. Les balises transitives persistent pendant le chaînage des rôles. Pour plus d'informations, veuillez consulter Chaînage des rôles avec des balises de session.

Pour définir les balises `Department` et `Project` comme transitives, utilisez l'attribut à valeurs multiples suivant :

Example Extrait d'une assertion SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>Department</AttributeValue>
</Attribute>
```

Transmission de balises de session en utilisant AssumeRoleWithWebIdentity

Utilisez une fédération conforme à OpenID Connect (OIDC) pour authentifier l'opération. `AssumeRoleWithWebIdentity` Cette opération renvoie un ensemble d'informations d'identification

temporaires que vous pouvez utiliser pour accéder aux AWS ressources. Pour plus d'informations sur l'utilisation de la fédération d'identité Web pour AWS Management Console l'accès, consultezFédération OIDC.

Pour transmettre les balises de session depuis OpenID Connect (OIDC), vous devez inclure les balises de session dans le JWT (JSON Web Token). Incluez les balises de session dans l'espace de noms `https://aws.amazon.com/ tags` dans le jeton lorsque vous soumettez la demande `AssumeRoleWithWebIdentity`. Pour en savoir plus sur les jetons OIDC et les revendications, veuillez consulter Utilisation des jetons avec les groupes d'utilisateurs dans le Guide du développeur Amazon Cognito .

Par exemple, le JWT décodé suivant est un jeton utilisé pour appeler `AssumeRoleWithWebIdentity` avec les balises de session `Project`, `CostCenter` et `Department`. Le jeton définit également les balises `Project` et `CostCenter` comme transitives. Les balises transitives persistent pendant le chaînage des rôles. Pour plus d'informations, veuillez consulter Chaînage des rôles avec des balises de session.

Example Exemple de jeton web JSON décodé

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags": {
        "principal_tags": {
            "Project": ["Automation"],
            "CostCenter": ["987654"],
            "Department": ["Engineering"]
        },
        "transitive_tag_keys": [
            "Project",
            "CostCenter"
        ]
    }
}
```

Transmission de balises de session en utilisant GetFederationToken

Le `GetFederationToken` vous permet de fédérer votre utilisateur. Cette opération renvoie un ensemble d'informations d'identification temporaires que vous pouvez utiliser pour accéder aux AWS ressources. Pour ajouter des balises à votre session utilisateur fédérée, utilisez l'`--tags` AWS CLI option ou le paramètre `Tags` AWS API. Vous ne pouvez pas définir les balises de session comme transitives lorsque vous utilisez le `GetFederationToken`, car vous ne pouvez pas utiliser les informations d'identification temporaires pour endosser un rôle. Vous ne pouvez pas utiliser le chaînage de rôles dans ce cas.

Voici un exemple de réponse à l'aide de `GetFederationToken`. Dans cet exemple, lorsque vous demandez le jeton, vous créez une session nommée `my-fed-user`. Vous ajoutez les paires clé-valeur de la balise de session `Project` = `Automation` et `Department` = `Engineering`.

Example Exemple de demande GetFederationToken CLI

```
aws sts get-federation-token \
--name my-fed-user \
--tags key=Project,value=Automation key=Department,value=Engineering
```

Lorsque vous utilisez les informations d'identification temporaires renvoyées par l'opération `GetFederationToken`, les balises principales de la session incluent les balises de l'utilisateur et les balises de session transmises.

Chaînage des rôles avec des balises de session

Vous pouvez endosser un rôle, puis utiliser les informations d'identification temporaires pour en endosser un autre. Vous pouvez continuer d'une session à une autre. C'est ce qu'on appelle le chaînage des rôles. Lorsque vous transmettez des balises de session en endossant un rôle, vous pouvez définir les clés comme transitives. Cela garantit que ces balises de session sont transmises aux sessions suivantes dans une chaîne de rôles. Vous ne pouvez pas définir les balises de rôle comme transitives. Pour transmettre ces balises aux sessions suivantes, indiquez-les en tant que balises de session.

> **Note**
>
> Les balises transitives persistent durant le chaînage des rôles et remplacent la mise en correspondance de valeurs `ResourceTag` après l'évaluation de la politique de confiance de rôle.

L'exemple suivant montre comment transmettre AWS STS les balises de session, les balises transitives et les balises de rôle aux sessions suivantes d'une chaîne de rôles.

Dans cet exemple de scénario de chaînage de rôles, vous utilisez une clé d'accès utilisateur IAM dans le AWS CLI pour assumer un rôle nommé. `Role1` Vous utilisez ensuite les informations d'identification de session résultantes pour endosser un second rôle nommé `Role2`. Vous pouvez ensuite utiliser les informations d'identification de la deuxième session pour endosser un troisième rôle nommé `Role3`. Ces demandes se déroulent sous la forme de trois opérations distinctes. Chaque rôle est déjà balisé dans IAM. Et lors de chaque demande, vous transmettez des balises de session supplémentaires.

Lorsque vous enchaînez des rôles, vous pouvez vous assurer que les balises d'une session antérieure persistent jusqu'aux sessions ultérieures. Pour ce faire à l'aide de la commande de la CLI `assume-role`, vous devez transmettre la balise en tant que balise de session et la définir comme transitive. Vous transmettez la balise `Star = 1` en tant que balise de session. La commande attache également la balise `Heart = 1` au rôle et s'applique en tant que balise principale lorsque vous utilisez la session. Cependant, vous voulez également que la balise `Heart = 1` transmette automatiquement à la deuxième ou troisième session. Pour ce faire, vous l'incluez manuellement en tant que balise de session. Les balises de principal de session résultantes incluent ces deux balises et les définissent comme transitives.

Vous effectuez cette demande à l'aide de la AWS CLI commande suivante :

Example Exemple de demande AssumeRole CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role1 \
--role-session-name Session1 \
--tags Key=Star,Value=1 Key=Heart,Value=1 \
--transitive-tag-keys Star Heart
```

Vous utilisez ensuite les informations d'identification pour cette session pour endosser `Role2`. La commande attache la balise `Sun = 2` au deuxième rôle et s'applique en tant que balise principale lorsque vous utilisez la deuxième session. Les balises `Heart = Star` héritent des balises de session transitive de la première session. Les balises principales résultantes de la deuxième session sont `Heart = 1`, `Star = 1` et `Sun = 2`. `Heart` et `Star` continueront d'être transitives. La balise `Sun` attachée à `Role2` n'est pas marquée comme transitive, car il ne s'agit pas d'une balise de session. Les sessions futures n'héritent pas de cette balise.

Vous effectuez cette deuxième demande à l'aide de la AWS CLI commande suivante :

Example Exemple de demande AssumeRole CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role2 \
--role-session-name Session2
```

Vous utilisez ensuite les informations d'identification de la deuxième session pour endosser `Role3`. Les balises principales de la troisième session proviennent de toutes les nouvelles balises de session, des balises de session transitive héritées et des balises de rôle. Les balises `Heart = 1` et `Star = 1` de la deuxième session ont été héritées de la balise de session transitive de la première session. Si vous essayez de transmettre la balise de session `Sun = 2`, l'opération échoue. La balise de session `Star = 1` héritée remplace la balise du rôle `Star = 3`. Dans le chaînage des rôles, la valeur d'une balise transitive remplace le rôle correspondant à la valeur `ResourceTag` avant l'évaluation de la politique de confiance de rôle. Dans cet exemple, si `Role3` utilise `Star` en tant que `ResourceTag` dans la politique de confiance de rôle, et définit la valeur `ResourceTag` à la valeur de balise transitive à partir de la session de rôle appelant. La balise du rôle `Lightning` s'applique également à la troisième session et n'est pas définie comme transitive.

Vous effectuez la troisième demande à l'aide de la AWS CLI commande suivante :

Example Exemple de demande AssumeRole CLI

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role3 \
--role-session-name Session3
```

Utilisation des balises de session pour ABAC

Le contrôle d'accès basé sur les attributs (ABAC) utilise une stratégie d'autorisation qui définit les autorisations en fonction des attributs de balise.

Si votre entreprise utilise un fournisseur d'identité (IdP) basé sur OIDC ou SAML pour gérer les identités des utilisateurs, vous pouvez configurer votre assertion pour transmettre les balises de session à AWS. Par exemple, dans le cas des identités d'utilisateur d'entreprise, lorsque vos employés se fédérent dans AWS, leurs attributs sont AWS appliqués au principal qui en résulte. Vous pouvez ensuite utiliser l'ABAC pour autoriser ou refuser des autorisations sur la base de ces attributs. Pour plus de détails, consultez Didacticiel IAM : utilisation de balises de session SAML pour le contrôle ABAC.

Pour plus d'informations sur l'utilisation d'IAM Identity Center avec ABAC, consultez la section Attributs pour le contrôle d'accès dans le Guide de l'utilisateur de AWS IAM Identity Center .

Afficher les balises de session dans CloudTrail

Vous pouvez l'utiliser AWS CloudTrail pour afficher les demandes utilisées pour assumer des rôles ou fédérer des utilisateurs. Le fichier CloudTrail journal contient des informations sur les principales balises de la session utilisateur assumée ou fédérée. Pour plus d'informations, consultez Journalisation des appels IAM et AWS STS API avec AWS CloudTrail.

Supposons, par exemple, que vous fassiez une AWS STS `AssumeRoleWithSAML` demande, que vous transmettiez des balises de session et que vous définissiez ces balises comme transitives. Vous trouverez les informations suivantes dans votre CloudTrail journal.

Example Exemple de AssumeRoleWith journal SAML CloudTrail

```
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyRoleSessionName",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "durationSeconds": 3600,
        "roleArn": "arn:aws:iam::123456789012:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::123456789012:saml-provider/Shibboleth"
    },
```

Vous pouvez consulter les exemples de CloudTrail journaux suivants pour visualiser les événements utilisant des balises de session.

- Exemple d'événement d'API de chaînage de AWS STS rôles dans un fichier CloudTrail journal
- Exemple d'événement d' AWS STS API SAML dans un fichier CloudTrail journal
- Exemple d'événement d' AWS STS API OIDC dans un fichier CloudTrail journal

Journalisation des appels IAM et AWS STS API avec AWS CloudTrail

IAM et AWS STS sont intégrés à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur ou un rôle IAM. CloudTrail capture tous les appels d'API pour IAM et AWS STS en tant qu'événements, y compris les appels depuis la console et depuis les appels d'API. Si vous créez un suivi, vous pouvez activer la diffusion continue des CloudTrail événements vers un compartiment Amazon S3. Si vous ne configurez pas de suivi, vous pouvez toujours consulter les événements les plus récents dans la CloudTrail console dans Historique des événements. Vous pouvez l'utiliser CloudTrail pour obtenir des informations sur la demande qui a été faite à IAM ou AWS STS. Par exemple, vous pouvez afficher l'adresse IP à partir de laquelle la demande a été effectuée, qui a effectué la demande et quand, ainsi que des détails supplémentaires.

Pour en savoir plus CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.

Rubriques

- IAM et AWS STS informations dans CloudTrail
- Journalisation des demandes IAM et AWS STS API
- Journalisation des demandes d'API vers d'autres services AWS
- Journalisation des événements de connexion utilisateur
- Journalisation des événements de connexion pour les informations d'identification temporaires
- Exemple d'événements d'API IAM dans le journal CloudTrail
- Exemples AWS STS d'événements d'API dans le CloudTrail journal
- Exemple d'événements de connexion dans le journal CloudTrail
- Comportement de la politique de confiance dans les rôles IAM

IAM et AWS STS informations dans CloudTrail

CloudTrail est activé sur votre compte Compte AWS lorsque vous créez le compte. Lorsqu'une activité se produit dans IAM ou AWS STS, cette activité est enregistrée dans un CloudTrail événement avec d'autres événements de AWS service dans l'historique des événements. Vous pouvez consulter, rechercher et télécharger les événements récents dans votre Compte AWS. Pour plus d'informations, consultez la section Affichage des événements à l'aide de l'historique des CloudTrail événements.

Pour un enregistrement continu des événements de votre entreprise Compte AWS, y compris les événements pour IAM AWS STS, et créez un parcours. Un suivi permet CloudTrail de fournir des fichiers journaux à un compartiment Amazon S3. Par défaut, lorsque vous créez un journal de suivi dans la console, il s'applique à toutes les régions . Le journal enregistre les événements de toutes les régions de la AWS partition et transmet les fichiers journaux au compartiment Amazon S3 que vous spécifiez. En outre, vous pouvez configurer d'autres AWS services pour analyser plus en détail les données d'événements collectées dans les CloudTrail journaux et agir en conséquence. section withinPour plus d'informations, consultez :

- Présentation de la création d'un journal d'activité
- CloudTrail Services et intégrations pris en charge
- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers CloudTrail journaux de plusieurs régions et réception de fichiers CloudTrail journaux de plusieurs comptes

Toutes les AWS STS actions et tous les IAM sont enregistrés CloudTrail et documentés dans les références d'API IAM et les références d'AWS Security Token Service API.

Journalisation des demandes IAM et AWS STS API

CloudTrail enregistre toutes les demandes d'API authentifiées dans les opérations IAM et AWS STS API. CloudTrail enregistre également les demandes non authentifiées relatives aux AWS STS actions`AssumeRoleWithWebIdentity`, `AssumeRoleWithSAML` et enregistre les informations fournies par le fournisseur d'identité. Cependant, certaines AWS STS demandes non authentifiées peuvent ne pas être enregistrées car elles ne répondent pas à l'attente minimale d'être suffisamment valides pour être considérées comme des demandes légitimes.

Vous pouvez utiliser les informations enregistrées pour mapper les appels passés par un utilisateur fédéré ayant un rôle assumé vers l'appelant fédéré externe d'origine. Dans ce cas`AssumeRole`, vous pouvez rediriger les appels vers le AWS service d'origine ou vers le compte de l'utilisateur d'origine. La `userIdentity` section des données JSON de l'entrée du CloudTrail journal contient les informations dont vous avez besoin pour associer la AssumeRole* demande à un utilisateur fédéré spécifique. Pour plus d'informations, consultez la section CloudTrail UserIdentity Element dans le guide de l'AWS CloudTrail utilisateur.

Par exemple, les appels à l'IAM`CreateUser`, `DeleteRoleListGroups`, et aux autres opérations d'API sont tous enregistrés par CloudTrail.

Des exemples de ce type d'entrée de journal sont présentés ultérieurement dans cette rubrique.

Journalisation des demandes d'API vers d'autres services AWS

Les demandes authentifiées adressées à d'autres opérations d'API de AWS service sont enregistrées CloudTrail, et ces entrées de journal contiennent des informations sur l'auteur de la demande.

Par exemple, supposons que vous ayez effectué une demande de liste des instances Amazon EC2 ou de création d'un groupe de déploiement AWS CodeDeploy . Les détails sur la personne ou le service qui a émis la demande sont enregistrés dans l'entrée de journal de cette demande. Ces informations vous aident à déterminer si la demande a été faite par un utilisateur IAM, un rôle ou un autre AWS service. Utilisateur racine d'un compte AWS

Pour plus de détails sur les informations d'identité utilisateur contenues dans les entrées du CloudTrail journal, consultez la section UserIdentity Element dans le guide de l'AWS CloudTrail utilisateur.

Journalisation des événements de connexion utilisateur

CloudTrail enregistre les événements de connexion aux AWS Management Console forums de AWS discussion et AWS Marketplace. CloudTrailenregistre les tentatives de connexion réussies et infructueuses pour les utilisateurs IAM et les utilisateurs fédérés.

Pour consulter CloudTrail des exemples d'événements relatifs à des connexions d'utilisateurs root réussies ou non, consultez la section Exemples d'enregistrements d'événements pour les utilisateurs root dans le Guide de l'AWS CloudTrail utilisateur.

Pour des raisons de sécurité, AWS n'enregistre pas le texte du nom d'utilisateur IAM saisi lorsque l'échec de connexion est dû à un nom d'utilisateur incorrect. Le texte du nom utilisateur est masqué par la valeur `HIDDEN_DUE_TO_SECURITY_REASONS`. Pour en obtenir un exemple, consultez Exemple d'événement d'échec de connexion provoqué par un nom d'utilisateur incorrect, plus loin dans cette rubrique. Le texte du nom d'utilisateur est obscurci car de tels échecs peuvent être causés par des erreurs d'utilisateur. La consignation de ces erreurs pourrait exposer des informations potentiellement sensibles. Par exemple :

- Vous tapez par erreur votre mot de passe dans le champ de nom d'utilisateur.
- Vous choisissez le lien vers la page de connexion de l'un d'entre eux Compte AWS, puis vous tapez le numéro de compte d'un autre Compte AWS.

- Vous oubliez sur quel compte vous vous connectez et vous tapez par erreur le nom de compte de votre compte de messagerie personnelle, votre identificateur de connexion bancaire ou un autre ID privé.

Journalisation des événements de connexion pour les informations d'identification temporaires

Lorsqu'un principal demande des informations d'identification temporaires, le type principal détermine la manière dont CloudTrail l'événement est enregistré. Cela peut être compliqué lorsqu'un principal endosse un rôle dans un autre compte. Il existe plusieurs appels d'API pour effectuer des opérations liées aux opérations inter-comptes de rôle. Tout d'abord, le principal appelle une AWS STS API pour récupérer les informations d'identification temporaires. Cette opération est enregistrée dans le compte d'appel et dans le compte sur lequel l' AWS STS opération est effectuée. Ensuite, le principal utilise le rôle pour effectuer d'autres appels d'API dans le compte du rôle endossé.

Vous pouvez utiliser la clé de condition `sts:SourceIdentity` dans la politique d'approbation de rôle pour exiger des utilisateurs qu'ils spécifient une identité lorsqu'ils endossent un rôle. Par exemple, vous pouvez exiger que les utilisateurs IAM spécifient leur propre nom d'utilisateur comme identité de source. Cela peut vous aider à déterminer quel utilisateur a effectué une action spécifique dans AWS. Pour plus d’informations, consultez `sts:SourceIdentity`. Vous pouvez utiliser `sts:RoleSessionName` pour exiger des utilisateurs qu'ils spécifient un nom de session lorsqu'ils endossent un rôle. Cela peut vous aider à différencier les sessions de rôle pour un rôle utilisé par différents principaux lorsque vous consultez les AWS CloudTrail journaux.

Le tableau suivant montre comment CloudTrail enregistre les différentes informations d'identité utilisateur pour chacune des AWS STS API qui génèrent des informations d'identification temporaires.

Type de principal	API STS	Identité de l'utilisateur dans le CloudTrai l journal du compte de l'appelant	Identité de l'utilisateur dans le CloudTrail journal pour le compte du rôle assumé	Identité de l'utilisateur dans le CloudTrail journal pour les appels d'API suivants du rôle
Utilisateur racine d'un compte	GetSessio nToken	Identité de l'utilisateur racine	Le compte du propriétaire de	Identité de l'utilisateur racine

Type de principal	API STS	Identité de l'utilisateur dans le CloudTrail journal du compte de l'appelant	Identité de l'utilisateur dans le CloudTrail journal pour le compte du rôle assumé	Identité de l'utilisateur dans le CloudTrail journal pour les appels d'API suivants du rôle
AWS informations d'identification			rôle est le même que le compte appelant	
Utilisateur IAM	GetSessionToken	Identité d'utilisateur IAM	Le compte du propriétaire de rôle est le même que le compte appelant	Identité d'utilisateur IAM
Utilisateur IAM	GetFederationToken	Identité d'utilisateur IAM	Le compte du propriétaire de rôle est le même que le compte appelant	Identité d'utilisateur IAM
Utilisateur IAM	AssumeRole	Identité d'utilisateur IAM	Numéro de compte et identifiant principal (s'il s'agit d'un utilisateur), ou principal AWS du service	Identité de rôle uniquement (aucun utilisateur)
Utilisateur authentifié en externe	AssumeRoleWithSAML	N/A	Identité d'utilisateur SAML	Identité de rôle uniquement (aucun utilisateur)

Type de principal	API STS	Identité de l'utilisateur dans le CloudTrail journal du compte de l'appelant	Identité de l'utilisateur dans le CloudTrail journal pour le compte du rôle assumé	Identité de l'utilisateur dans le CloudTrail journal pour les appels d'API suivants du rôle
Utilisateur authentifié en externe	AssumeRoleWithWebIdentity	N/A	Identité d'utilisateur OIDC/web	Identité de rôle uniquement (aucun utilisateur)

CloudTrail considère une action en lecture seule si elle n'a aucun effet mutant sur une ressource. Lorsque vous enregistrez un événement en lecture seule, CloudTrail supprimez les `responseElements` informations du journal. Lorsque vous CloudTrail enregistrez un événement qui n'est pas en lecture seule, l'intégralité `responseElements` est affichée dans l'entrée du journal. Toutefois, pour les AWS STS API `AssumeRole` `AssumeRoleWithSAMLAssumeRoleWithWebIdentity`, et même si elles sont enregistrées en lecture seule, l'intégralité de ces API CloudTrail sera incluse `responseElements` dans le journal.

Le tableau suivant indique le mode de CloudTrail journalisation `responseElements` et les `readOnly` informations de chacune des AWS STS API qui génèrent des informations d'identification temporaires.

API STS	Informations sur les éléments de réponse	Lecture seule
AssumeRole	Inclus	true
AssumeRoleWithSAML	Inclus	true
AssumeRoleWithWebIdentity	Inclus	true
GetFederationToken	Inclus	false
GetSessionToken	Inclus	false

Exemple d'événements d'API IAM dans le journal CloudTrail

CloudTrail les fichiers journaux contiennent des événements formatés à l'aide de JSON. Un événement d'API représente une demande d'API unique et inclut des informations sur l'action principale demandée, sur tous les paramètres, et sur la date et l'heure de l'action.

Exemple d'événement d'API IAM dans un fichier CloudTrail journal

L'exemple suivant montre une entrée de CloudTrail journal pour une demande effectuée pour l'`GetUserPolicy`action IAM.


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/JaneDoe",
    "accountId": "444455556666",
    "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
    "userName": "JaneDoe",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2014-07-15T21:39:40Z"
      }
    },
    "invokedBy": "signin.amazonaws.com"
  },
  "eventTime": "2014-07-15T21:40:14Z",
  "eventSource": "iam.amazonaws.com",
  "eventName": "GetUserPolicy",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "signin.amazonaws.com",
  "userAgent": "signin.amazonaws.com",
  "requestParameters": {
    "userName": "JaneDoe",
    "policyName": "ReadOnlyAccess-JaneDoe-201407151307"
  },
  "responseElements": null,
  "requestID": "9EXAMPLE-0c68-11e4-a24e-d5e16EXAMPLE",
  "eventID": "cEXAMPLE-127e-4632-980d-505a4EXAMPLE"
}
```

À partir de ces informations d'événement, vous pouvez déterminer que la demande a été effectuée pour obtenir une politique utilisateur nommée `ReadOnlyAccess-JaneDoe-201407151307` pour l'utilisateur `JaneDoe`, comme indiqué dans l'élément `requestParameters`. Vous pouvez également voir que la demande a été effectuée par une utilisatrice IAM nommée `JaneDoe` le 15 juillet 2014 à 21:40 (UTC). Dans ce cas, la demande provient du AWS Management Console, comme vous pouvez le constater à partir de l'`userAgent`élément.

Exemples AWS STS d'événements d'API dans le CloudTrail journal

CloudTrail les fichiers journaux contiennent des événements formatés à l'aide de JSON. Un événement d'API représente une demande d'API unique et inclut des informations sur l'action principale demandée, sur tous les paramètres, et sur la date et l'heure de l'action.

Exemples d'événements d' AWS STS API entre comptes dans des fichiers CloudTrail journaux

L'utilisateur IAM nommé `JohnDoe` dans le compte 777788889999 lance l' AWS STS `AssumeRole`action pour assumer le rôle dans le compte 111122223333. `EC2-dev` L'administrateur de compte exige des utilisateurs qu'ils définissent une identité source équivalente à leur nom d'utilisateur lorsqu'ils endossent le rôle. L'utilisateur transmet la valeur d'identité source de `JohnDoe`.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "arn": "arn:aws:iam::777788889999:user/JohnDoe",
    "accountId": "777788889999",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "JohnDoe"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "JohnDoe-EC2-dev",
```

```
    "sourceIdentity": "JohnDoe",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2023, 4:07:39 PM"
      },
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:JohnDoe-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/JohnDoe-EC2-dev"
    },
  "sourceIdentity": "JohnDoe"
  },
  "resources": [
    {
      "ARN": "arn:aws:iam::111122223333:role/EC2-dev",
      "accountId": "111122223333",
      "type": "AWS::IAM::Role"
    }
  ],
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

Le deuxième exemple montre l'entrée de CloudTrail journal du compte de rôle supposé (111122223333) pour la même demande.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AWSAccount",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "accountId": "777788889999"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
```

```
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "JohnDoe-EC2-dev",
    "sourceIdentity": "JohnDoe",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2014, 4:07:39 PM"
    },
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:JohnDoe-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/JohnDoe-EC2-dev"
      },
  "sourceIdentity": "JohnDoe"
  },
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE"
}
```


Exemple d'événement d'API de chaînage de AWS STS rôles dans un fichier CloudTrail journal

L'exemple suivant montre une entrée de CloudTrail journal pour une demande faite par John Doe dans le compte 111111111111. John a précédemment utilisé son utilisateur `JohnDoe` pour endosser le rôle `JohnRole1`. Pour cette demande, il utilise les informations d'identification de ce rôle pour endosser le rôle `JohnRole2`. Ceci est connu sous le nom de chaînage de rôles. L'identité source qu'il a définie lorsqu'il a endossé le rôle `JohnDoe1` persiste dans la demande d'endosser le `JohnRole2`. Si John tente de définir une identité source différente lorsqu'il endosse le rôle, la demande est refusée. John transmet deux balises de session dans la demande. Il définit ces deux balises comme transitives. La demande hérite de la balise `Department` comme transitive car John l'a définie comme transitive lorsqu'il a endossé `JohnRole1`. Pour plus d'informations sur l'identité source, consultez Surveiller et contrôler les actions prises avec les rôles endossés. Pour de plus amples informations sur les clés transitives dans les chaînes de rôles, veuillez consulter Chaînage des rôles avec des balises de session.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIN5ATK5U7KEXAMPLE:JohnRole1",
        "arn": "arn:aws:sts::111111111111:assumed-role/JohnDoe/JohnRole1",
        "accountId": "111111111111",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2019-10-02T21:50:54Z"
            },
                "sessionIssuer": {
                "type": "Role",
               "principalId": "AROAIN5ATK5U7KEXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/JohnRole1",
                "accountId": "111111111111",
                "userName": "JohnDoe"
            },
            "sourceIdentity": "JohnDoe"
        }
    },
    "eventTime": "2019-10-02T22:12:29Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "123.145.67.89",
    "userAgent": "aws-cli/1.16.248 Python/3.4.7
 Linux/4.9.184-0.1.ac.235.83.329.metal1.x86_64 botocore/1.12.239",
    "requestParameters": {
        "incomingTransitiveTags": {
            "Department": "Engineering"
        },
        "tags": [
            {
                "value": "johndoe@example.com",
                "key": "Email"
            },
            {
                "value": "12345",
                "key": "CostCenter"
            }
```

```
        ],
        "roleArn": "arn:aws:iam::111111111111:role/JohnRole2",
        "roleSessionName": "Role2WithTags",
        "sourceIdentity": "JohnDoe",
        "transitiveTagKeys": [
            "Email",
            "CostCenter"
        ],
        "durationSeconds": 3600
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
            "expiration": "Oct 2, 2019, 11:12:29 PM",
            "sessionToken": "AgoJb3JpZ2luX2VjEB4aCXVzLXdlc3QtMSJHMEXAMPLETOKEN
+//rJb8Lo30mFc5MlhFCEbubZvEj0wHB/mDMwIgSEe9gk/Zjr09tZV7F1HDTMhmEXAMPLETOKEN/iEJ/
rkqngII9//////////
ARABGgw0MjgzMDc4NjM5NjYiDLZjZFKwP4qxQG5sFCryASO4UPz5qE97wPPH1eLMvs7CgSDBSWfonmRTCfokm2FN1+hWUdQ
+C+WKFZb701eiv9J5La2EXAMPLETOKEN/c7S5Iro1WUJ0q3Cxuo/8HUoSxVhQHM7zF7mWWLhXLEQ52ivL
+F6q5dpXu4aTFedpMfnJa8JtkWwG9x1Axj0Ypy2ok8v5unpQGWych1vwdvj6ez1Dm8Xg1+qIzXILiEXAMPLETOKEN/
vQGqu8H+nxp3kabcrtOvTFTvxX6vsc8OGwUfHhzAfYGEXAMPLETOKEN/
L6v1yMM3B1OwFOrQBno1HEjf1oNI8RnQiMNFdUOtwYj7HUZIOCZmjfN8PPHq77N7GJl9lzvIZKQA0Owcjg
+mc78zHCj8y0siY8C96paEXAMPLETOKEN/
E3cpksxWdgs91HRzJWScjN2+r2LTGjYhyPqcmFzzo2mCE7mBNEXAMPLETOKEN/oJy
+2o83YNW5tOiDmczgDzJZ4UKR84yGYOMfSnF4XcEJrDgAJ3OJFwmTcTQICAlSwLEXAMPLETOKEN"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROAIFR7WHDTSOYQYHFUE:Role2WithTags",
            "arn": "arn:aws:sts::111111111111:assumed-role/test-role/Role2WithTags"
        },
    "sourceIdentity": "JohnDoe"
    },
    "requestID": "b96b0e4e-e561-11e9-8b3f-7b396EXAMPLE",
    "eventID": "1917948f-3042-46ec-98e2-62865EXAMPLE",
    "resources": [
        {
            "ARN": "arn:aws:iam::111111111111:role/JohnRole2",
            "accountId": "111111111111",
            "type": "AWS::IAM::Role"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111111111111"
```

```
}
```

Exemple d'événement AWS STS d'API de AWS service dans le fichier CloudTrail journal

L'exemple suivant montre une entrée de CloudTrail journal pour une demande faite par un AWS service appelant une autre API de service à l'aide des autorisations d'un rôle de service. Il affiche l'entrée du CloudTrail journal pour la demande effectuée dans le compte 777788889999.

```
{
  "eventVersion": "1.04",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAQRSTUVWXYZEXAMPLE:devdsk",
    "arn": "arn:aws:sts::777788889999:assumed-role/AssumeNothing/devdsk",
    "accountId": "777788889999",
    "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2016-11-14T17:25:26Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAQRSTUVWXYZEXAMPLE",
        "arn": "arn:aws:iam::777788889999:role/AssumeNothing",
        "accountId": "777788889999",
        "userName": "AssumeNothing"
      }
    }
  },
  "eventTime": "2016-11-14T17:25:45Z",
  "eventSource": "s3.amazonaws.com",
  "eventName": "DeleteBucket",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "[aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67]",
  "requestParameters": {
    "bucketName": "my-test-bucket-cross-account"
  },
  "responseElements": null,
  "requestID": "EXAMPLE463D56D4C",
```

```
  "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "777788889999"
}
```

Exemple d'événement d' AWS STS API SAML dans un fichier CloudTrail journal

L'exemple suivant montre une entrée de CloudTrail journal pour une demande faite pour l' AWS STS `AssumeRoleWithSAML`action. La demande inclut les attributs SAML `CostCenter` et `Project` qui sont transmis par l'assertion SAML en tant que balises de session. Ces balises sont définies comme transitives afin qu'elles persistent dans les scénarios de chaînage des rôles. La demande inclut le paramètre d'API facultatif`DurationSeconds`, représenté `durationSeconds` dans le CloudTrail journal, et est définie sur `1800` secondes. La demande inclut également l'attribut SAML `sourceIdentity`, qui est transmise dans l'assertion SAML. Si quelqu'un utilise les informations d'identification de session de rôle résultantes pour endosser un autre rôle, cette identité source persiste.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "SAMLUser",
        "principalId": "SampleUkh1i4+ExamplexL/jEvs=:SamlExample",
        "userName": "SamlExample",
        "identityProvider": "bdGOnTesti4+ExamplexL/jEvs="
    },
    "eventTime": "2023-08-28T18:30:58Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoleWithSAML",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "aws-internal/3 aws-sdk-java/1.12.479
 Linux/5.10.186-157.751.amzn2int.x86_64 OpenJDK_64-Bit_Server_VM/17.0.7+11 java/17.0.7
 kotlin/1.3.72 vendor/Amazon.com_Inc. cfg/retry-mode/standard",
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyAssignedRoleSessionName",
        "sourceIdentity": "MySAMLUser",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn",
            "Department": "Engineering"
        },
```

```
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "roleArn": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::444455556666:saml-provider/Shibboleth",
        "durationSeconds": 1800
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Aug 28, 2023, 7:00:58 PM"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROAD35QRSTUVWEXAMPLE:MyAssignedRoleSessionName",
            "arn": "arn:aws:sts::444455556666:assumed-role/SAMLTestRoleShibboleth/
MyAssignedRoleSessionName"
        },
        "packedPolicySize": 1,
        "subject": "SamlExample",
        "subjectType": "transient",
        "issuer": "https://server.example.com/idp/shibboleth",
        "audience": "https://signin.aws.amazon.com/saml",
        "nameQualifier": "bdGOnTesti4+ExamplexL/jEvs=",
        "sourceIdentity": "MySAMLUser"
    },
    "requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
    "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth"
        },
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::SAMLProvider",
            "ARN": "arn:aws:iam::444455556666:saml-provider/test-saml-provider"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
```

```
    "recipientAccountId": "444455556666",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "sts.us-east-2.amazonaws.com"
    }
}
```

Exemple d'événement d' AWS STS API OIDC dans un fichier CloudTrail journal

L'exemple suivant montre une entrée de CloudTrail journal pour une demande faite pour l' AWS STS `AssumeRoleWithWebIdentity`action. La demande inclut les attributs `CostCenter` et `Project` qui sont transmis par le jeton du fournisseur d'identité en tant que balises de session. Ces balises sont définies comme transitives afin qu'elles persistent dans le chaînage des rôles. La demande inclut l'attribut `sourceIdentity` du jeton du fournisseur d'identité. Si quelqu'un utilise les informations d'identification de session de rôle résultantes pour endosser un autre rôle, cette identité source persiste.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "WebIdentityUser",
    "principalId": "accounts.google.com:<id-of-
application>.apps.googleusercontent.com:<id-of-user>",
    "userName": "<id of user>",
    "identityProvider": "accounts.google.com"
  },
  "eventTime": "2016-03-23T01:39:51Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRoleWithWebIdentity",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.3.23 Python/2.7.6 Linux/2.6.18-164.el5",
  "requestParameters": {
    "sourceIdentity": "MyWebIdentityUser",
    "durationSeconds": 3600,
    "roleArn": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
    "roleSessionName": "MyAssignedRoleSessionName"
        "principalTags": {
            "CostCenter": "24680",
            "Project": "Pegasus"
```

```
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
  },
  "responseElements": {
    "provider": "accounts.google.com",
    "subjectFromWebIdentityToken": "<id of user>",
    "sourceIdentity": "MyWebIdentityUser",
    "audience": "<id of application>.apps.googleusercontent.com",
    "credentials": {
      "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
      "expiration": "Mar 23, 2016, 2:39:51 AM",
      "sessionToken": "<encoded session token blob>"
    },
    "assumedRoleUser": {
      "assumedRoleId": "AROACQRSTUVWRAOEXAMPLE:MyAssignedRoleSessionName",
      "arn": "arn:aws:sts::444455556666:assumed-role/FederatedWebIdentityRole/
MyAssignedRoleSessionName"
    }
  },
  "resources": [
    {
      "ARN": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
      "accountId": "444455556666",
      "type": "AWS::IAM::Role"
    }
  ],
  "requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
  "eventID": "bEXAMPLE-0b30-4246-b28c-e3da3EXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "444455556666"
}
```

Exemple d'événements de connexion dans le journal CloudTrail

CloudTrail les fichiers journaux contiennent des événements formatés à l'aide de JSON. Un événement de connexion représente une seule demande de connexion et inclut des informations sur le principal de connexion, la région, ainsi que la date et l'heure de l'action.

Exemple d'événement de réussite de connexion dans le fichier CloudTrail journal

L'exemple suivant montre une entrée de CloudTrail journal indiquant un événement de connexion réussi.


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/JohnDoe",
    "accountId": "111122223333",
    "userName": "JohnDoe"
  },
  "eventTime": "2014-07-16T15:49:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.110",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Success"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/s3/  ",
    "MFAUsed": "No"
  },
  "eventID": "3fcfb182-98f8-4744-bd45-10a395ab61cb"
}
```


Pour plus de détails sur les informations contenues dans les fichiers CloudTrail journaux, consultez la section Référence des CloudTrail événements dans le guide de AWS CloudTrail l'utilisateur.

Exemple d'échec de connexion dans le fichier CloudTrail journal

L'exemple suivant montre une entrée de CloudTrail journal pour un échec de connexion.


```
{
  "eventVersion": "1.05",
```

```
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/JaneDoe",
    "accountId": "111122223333",
    "userName": "JaneDoe"
  },
  "eventTime": "2014-07-08T17:35:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.100",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "Failed authentication",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/sns",
    "MFAUsed": "No"
  },
  "eventID": "11ea990b-4678-4bcd-8fbe-62509088b7cf"
}
```

À partir de ces informations, vous pouvez déterminer que la tentative de connexion a été effectuée par une utilisatrice IAM nommée `JaneDoe`, comme illustré dans l'élément `userIdentity`. Vous pouvez également voir que la tentative de connexion a échoué, comme montré dans l'élément `responseElements`. Vous pouvez voir que `JaneDoe` a essayé de se connecter à la console Amazon SNS à 17:35 (UTC) le 8 juillet 2014.

Exemple d'événement d'échec de connexion provoqué par un nom d'utilisateur incorrect

L'exemple suivant montre une entrée de CloudTrail journal concernant un échec de connexion dû à la saisie d'un nom d'utilisateur incorrect par l'utilisateur. AWS masque le `userName` texte `HIDDEN_DUE_TO_SECURITY_REASONS` pour éviter d'exposer des informations potentiellement sensibles.

```
{
```

```
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "accountId": "123456789012",
    "accessKeyId": "",
    "userName": "HIDDEN_DUE_TO_SECURITY_REASONS"
  },
  "eventTime": "2015-03-31T22:20:42Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "No username found in supplied account",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs
%23&isauthcode=true",
    "MobileVersion": "No",
    "MFAUsed": "No"
  },
  "eventID": "a7654656-0417-45c6-9386-ea8231385051",
  "eventType": "AwsConsoleSignin",
  "recipientAccountId": "123456789012"
}
```


Comportement de la politique de confiance dans les rôles IAM

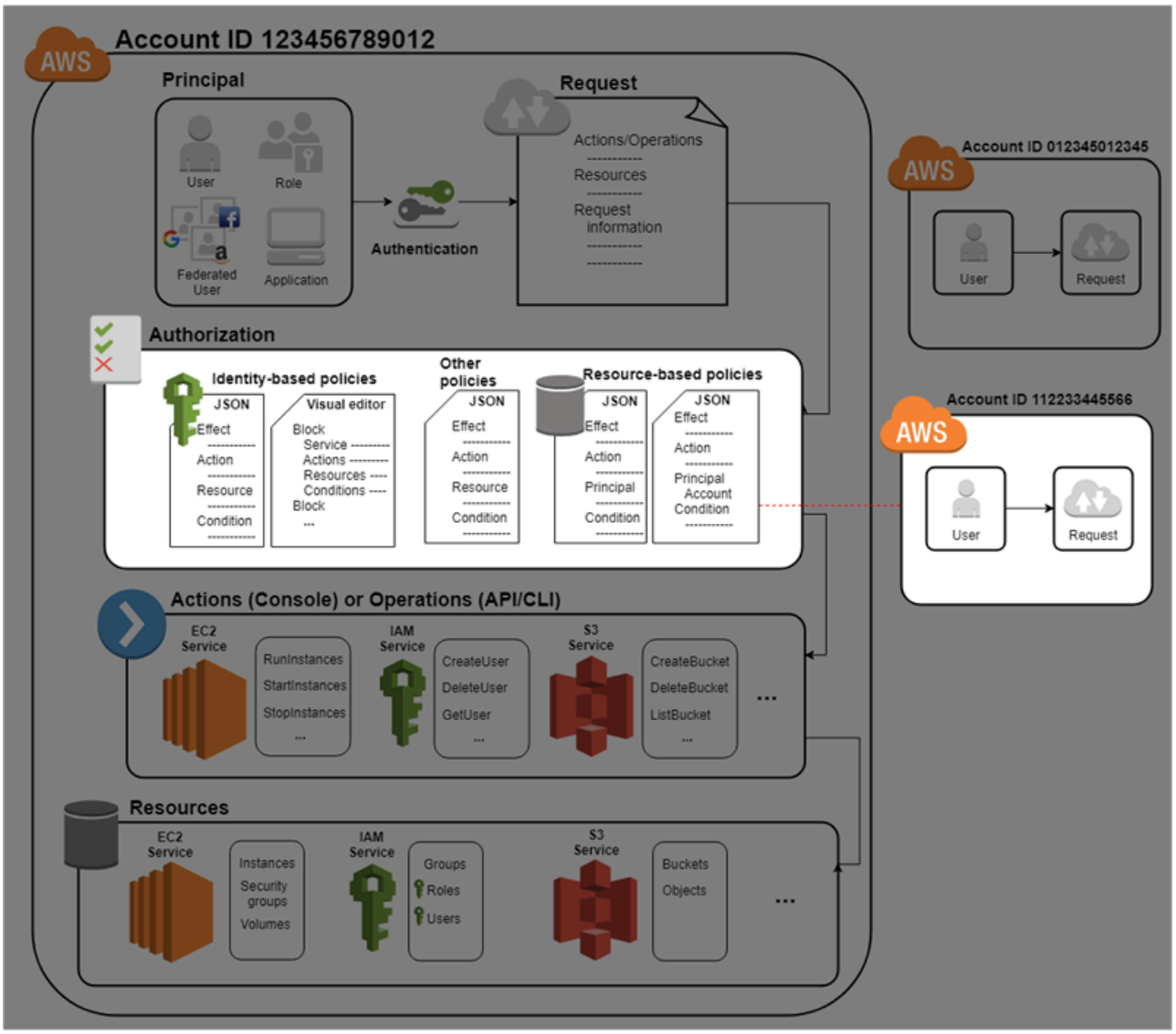
Le 21 septembre 2022, des modifications AWS ont été apportées au comportement de la politique de confiance des rôles IAM afin d'exiger des autorisations explicites dans une politique de confiance des rôles lorsqu'un rôle assume lui-même. Les rôles IAM figurant dans l'ancienne liste d'autorisation des comportements comportent un additionalEventData champ explicitTrustGrant destiné aux `AssumeRole` événements. La valeur de `explicitTrustGrant` est fausse lorsqu'un rôle figurant dans l'ancienne liste d'autorisation suppose qu'il utilise le comportement existant. Lorsqu'un rôle figurant dans l'ancienne liste d'autorisation assume lui-même son rôle mais que le comportement de la politique d'approbation des rôles a été mis à jour pour autoriser explicitement le rôle à s'assumer lui-même, la valeur de `explicitTrustGrant` est vraie.

Seul un très petit nombre de rôles IAM figurent sur la liste des rôles autorisés pour le comportement existant, et ce champ n'est présent dans les CloudTrail journaux de ces rôles que lorsqu'ils se reprennent eux-mêmes. Dans la plupart des cas, il n'est pas nécessaire qu'un rôle IAM assume lui-même. AWS recommande de mettre à jour vos processus, votre code ou vos configurations pour supprimer ce comportement ou de mettre à jour vos politiques d'approbation des rôles pour autoriser explicitement ce comportement. Pour plus d'informations, voir Annonce d'une mise à jour du comportement de la politique de confiance dans les rôles IAM.

Gestion de l'accès aux AWS ressources

AWS Identity and Access Management (IAM) est un service Web qui vous permet de contrôler en toute sécurité l'accès aux AWS ressources. Lorsqu'un principal fait une demande AWS, le code d' AWS exécution vérifie si le principal est authentifié (connecté) et autorisé (possède des autorisations). Vous gérez l'accès en AWS créant des politiques et en les associant aux identités ou aux AWS ressources IAM. Les politiques sont des documents JSON AWS qui, lorsqu'ils sont attachés à une identité ou à une ressource, définissent leurs autorisations. Pour plus d'informations sur les types de politiques et les utilisations, consultez Politiques et autorisations dans IAM.

Pour en savoir plus sur le reste du processus d'authentification et d'autorisation, consultez Fonctionnement de IAM.



Lors de l'autorisation, le code d' AWS application utilise les valeurs du contexte de la demande pour vérifier les politiques correspondantes et déterminer s'il convient d'autoriser ou de refuser la demande.

AWS vérifie chaque politique qui s'applique au contexte de la demande. Si une seule politique refuse la demande, AWS refuse l'intégralité de la demande et arrête l'évaluation des politiques. Ceci est appelé un refus explicite. Étant donnée que les demandes sont refusées par défaut, IAM autorise votre demande uniquement si chacune de ses parties est autorisée par les politiques applicables. La logique d'évaluation pour une demande au sein d'un même compte utilise les règles suivantes :

- Par défaut, toutes les demandes sont implicitement refusées. (Autrement, par défaut, le Utilisateur racine d'un compte AWS dispose d'un accès complet.)
- Une autorisation explicite dans une politique basée sur l'identité ou les ressources remplace cette valeur par défaut.
- Si une limite d'autorisations, SCP Organizations ou politique de session est présente, elle peut remplacer l'autorisation avec un refus implicite.
- Un refus explicite dans n'importe quelle stratégie remplace toutes les autorisations.

Une fois que votre demande a été authentifiée et autorisée, AWS approuve la demande. Si vous avez besoin d'effectuer une demande dans un compte différent, une politique dans l'autre compte doit vous permettre d'accéder à la ressource. En outre, l'entité IAM que vous utilisez pour effectuer la demande doit avoir une politique basée sur l'identité qui autorise la demande.

Ressources de gestion des accès

Pour plus d'informations sur les autorisations et la création de politiques, consultez les ressources suivantes :

Les entrées suivantes du blog sur la AWS sécurité décrivent les méthodes courantes de rédaction de politiques d'accès aux compartiments et aux objets Amazon S3.

- Writing IAM Policies: How to Grant Access to an Amazon S3 Bucket
- Writing IAM policies: Grant Access to User-Specific Folders in an Amazon S3 Bucket
- IAM Policies and Bucket Policies and ACLs! Oh My! (Contrôler l'accès aux ressources S3)
- A Primer on RDS Resource-Level Permissions
- Demystifying EC2 Resource-Level Permissions

Politiques et autorisations dans IAM

Vous gérez l'accès en AWS créant des politiques et en les associant à des identités IAM (utilisateurs, groupes d'utilisateurs ou rôles) ou à des AWS ressources. Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit leurs autorisations. AWS évalue ces politiques lorsqu'un principal IAM (utilisateur ou rôle) fait une demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée. La plupart des politiques sont stockées AWS sous forme de documents JSON. AWS prend en charge six types de politiques : les politiques basées sur l'identité, les politiques basées sur les ressources, les limites d'autorisations, les SCP des organisations, les ACL et les politiques de session.

Les politiques IAM définissent les autorisations d'une action, quelle que soit la méthode que vous utilisez pour exécuter l'opération. Par exemple, si une politique autorise l'GetUseraction, un utilisateur utilisant cette politique peut obtenir des informations utilisateur auprès de AWS Management Console, de AWS CLI, ou de l' AWS API. Lorsque vous créez un utilisateur IAM, vous pouvez choisir d'autoriser la console ou un accès par programme. Si l'accès à la console est autorisé, l'utilisateur IAM peut s'y connecter à l'aide de ses informations d'identification. Si l'accès programmatique est autorisé, l'utilisateur peut utiliser des clés d'accès pour travailler avec la CLI ou l'API.

Types de politique

Les types de politiques suivants sont répertoriés dans l'ordre du plus fréquemment utilisé au moins fréquemment utilisé, et ils peuvent être utilisés dans AWS. Pour de plus amples informations, veuillez consulter les sections ci-dessous pour chaque type de politique.

- Identity-based policies (Politiques basées sur l'identité) : attachez des politiques gérées et en ligne à des identités IAM (utilisateurs, groupes auxquels appartiennent des utilisateurs ou rôles). Les politiques basées sur l'identité accordent des autorisations à une identité.
- Resource-based policies (Politiques basées sur les ressources) : attachez des politiques en ligne aux ressources. Les exemples les plus courants de politiques basées sur les ressources sont les politiques de compartiment Amazon S3 et les politiques confiance de rôle IAM. Les politiques basées sur des ressources accordent des autorisations au principal qui est spécifié dans la politique. Les principaux peuvent être dans le même compte que la ressource ou dans d'autres comptes.
- Permissions boundaries (Limites d'autorisations) : utilisez une politique gérée en tant que limite d'autorisations pour une entité IAM (utilisateur ou rôle). Cette politique définit les autorisations maximales que les politiques basées sur une identité peuvent accorder à une entité, mais

n'accorde pas d'autorisations. Les limites d'autorisations ne définissent pas les autorisations maximales qu'une politique basée sur les ressources peut accorder à une entité.

- Organisations SCP : utilisez une politique de contrôle des AWS Organizations services (SCP) pour définir les autorisations maximales pour les membres du compte d'une organisation ou d'une unité organisationnelle (UO). Les politiques de contrôle des services limitent les autorisations que les politiques basées sur l'identité ou sur une ressource accordent à des entités (utilisateurs ou rôles) au sein du compte, mais n'accordent pas d'autorisations.
- Access control lists (ACLs) (Listes de contrôle d'accès [ACL]) : utilisez les listes de contrôle d'accès (ACL) pour contrôler quels principaux dans d'autres comptes peuvent accéder à la ressource à laquelle la liste de contrôle d'accès (ACL) est attachée. Les listes de contrôle d'accès sont semblables aux politiques basées sur les ressources, bien qu'elles soient le seul type de politique qui n'utilise pas la structure d'un document de politique JSON. Les listes de contrôle d'accès sont des politiques d'autorisations entre comptes qui accordent des autorisations pour le principal spécifié. Les listes de contrôle d'accès ne peuvent pas accorder des autorisations aux entités au sein du même compte.
- Politiques de session — Adoptez des politiques de session avancées lorsque vous utilisez l' AWS API AWS CLI or pour assumer un rôle ou un utilisateur fédéré. Les politiques de session limitent les autorisations que les politiques basées sur l'identité du rôle ou de l'utilisateur accordent à la session. Les politiques de session limitent les autorisations d'une session créée, mais n'accordent pas d'autorisations. Pour plus d'informations, consultez Politiques de session.

Politiques basées sur l'identité

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON qui contrôlent les actions qu'une identité (utilisateurs, groupes d'utilisateurs et rôles) peut effectuer, sur quelles ressources, et dans quelles conditions. Les politiques basées l'identité peuvent être classées comme suit :

- Politiques gérées : politiques autonomes basées sur l'identité que vous pouvez associer à plusieurs utilisateurs, groupes et rôles au sein de votre. Compte AWS Il existe deux types de politiques gérées.
  - AWS politiques gérées : politiques gérées créées et gérées par AWS.
  - Politiques gérées par le client : politiques gérées que vous créez et gérez dans votre Compte AWS. Les politiques gérées par le client fournissent un contrôle plus précis de vos politiques que les politiques AWS gérées.

- Politiques en ligne : politiques que vous ajoutez directement à un utilisateur unique, un groupe d'utilisateurs, ou un rôle. Les politiques intégrées maintiennent une one-to-one relation stricte entre une politique et une identité. Elles sont supprimées lorsque vous supprimez l'identité.

Pour savoir comment choisir entre des politiques gérées et en ligne, veuillez consulter Choix entre les politiques gérées et les politiques en ligne.

Politiques basées sur les ressources

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource, telle qu'un compartiment Amazon S3. Ces politiques accordent au principal spécifié l'autorisation d'effectuer des actions spécifiques sur cette ressource et définit sous quelles conditions cela s'applique. Les politiques basées sur les ressources sont des politiques en ligne. Il ne s'agit pas de politiques gérées basées sur les ressources.

Pour permettre un accès comptes multiples , vous pouvez spécifier un compte entier ou des entités IAM dans un autre compte en tant que principal dans une politique basée sur les ressources. L'ajout d'un principal entre comptes à une politique basée sur les ressources ne représente qu'une partie de l'instauration de la relation d'approbation. Lorsque le principal et la ressource sont séparés Comptes AWS, vous devez également utiliser une politique basée sur l'identité pour accorder au principal l'accès à la ressource. Toutefois, si une politique basée sur des ressources accorde l'accès à un principal dans le même compte, aucune autre politique basée sur l'identité n'est requise. Pour obtenir des instructions détaillées sur l'octroi d'un accès entre services, veuillez consulter Didacticiel IAM : déléguer l'accès entre des comptes AWS à l'aide des rôles IAM.

Le service IAM prend en charge un seul type de politique basée sur les ressources, nommé politique d'approbation de rôle, qui est attaché à un rôle IAM. Un rôle IAM est à la fois une identité et une ressource qui prend en charge les politiques basées sur les ressources. C'est pour cette raison que vous devez associer une politique d'approbation et une politique basée sur une identité à un rôle IAM. Les politiques d'approbation définissent quelles entités principaux (comptes, utilisateurs, rôles et utilisateurs fédérés) peuvent endosser le rôle. Pour en savoir plus sur la façon dont les rôles IAM sont différents d'autres politiques basées sur les ressources, consultez Accès intercompte aux ressources dans IAM.

Pour connaître les autres services qui prennent en charge les politiques basées sur les ressources, voir AWS services qui fonctionnent avec IAM. Pour en savoir plus sur politiques basées sur les ressources, voir Politiques basées sur l'identité et Politiques basées sur une ressource. Pour savoir si

les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, consultez Qu'est-ce que l'Analyseur d'accès IAM ?.

Limites d'autorisations IAM

Une limite d'autorisations est une fonctionnalité avancée dans laquelle vous définissez les autorisations maximales qu'une politique basée sur les identités peut accorder à une entité IAM. Lorsque vous définissez une limite d'autorisations pour une entité, l'entité peut effectuer uniquement les actions autorisées par ses deux ses stratégies basées sur l'identité et ses limites d'autorisations. Les politiques basées sur les ressources qui spécifient l'utilisateur ou le rôle en tant que principal ne sont pas limitées par les limites d'autorisations. Un refus explicite dans l’une de ces politiques annule l’autorisation. Pour plus d'informations sur les limites d'autorisations, consultez Limites d'autorisations pour les entités IAM.

Politiques de contrôle de service (SCP)

AWS Organizations est un service permettant de regrouper et de gérer de manière centralisée Comptes AWS les actifs détenus par votre entreprise. Si vous activez toutes les fonctionnalités d’une organisation, vous pouvez appliquer les politiques de contrôle des services (SCP) à l’un ou à l’ensemble de vos comptes. Les stratégies de contrôle de service qui spécifient les autorisations maximales pour une organisation ou une unité d'organisation. Le SCP limite les autorisations pour les entités figurant dans les comptes des membres, y compris chacune Utilisateur racine d'un compte AWS d'entre elles. Un refus explicite dans l’une de ces politiques annule l’autorisation.

Pour plus d'informations sur les organisations et les SCP, veuillez consulter Fonctionnement des SCP dans le Guide de l'utilisateur AWS Organizations .

Listes de contrôle d’accès (ACL)

Les listes de contrôle d'accès (ACL) sont des politiques de service qui vous permettent de contrôler quels principaux d'un autre compte peuvent accéder à une ressource. Les ACL ne peuvent pas être utilisées pour contrôler l'accès pour un principal au sein du même compte. Les listes de contrôle d'accès sont semblables aux politiques basées sur les ressources, bien qu'elles soient le seul type de politique qui n'utilise pas le format de document de politique JSON. Amazon S3 et Amazon VPC sont des exemples de services qui prennent en charge les ACL. AWS WAF Pour en savoir plus sur les listes de contrôle d'accès, veuillez consulter Présentation des listes de contrôle d'accès (ACL) dans le Guide du développeur Amazon Simple Storage Service.

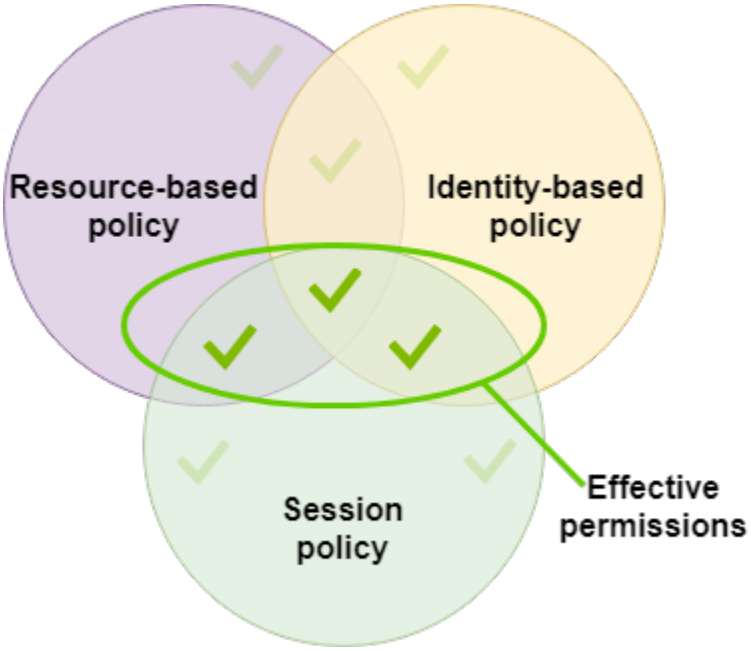
Politiques de session

Les politiques de session sont des politiques avancées que vous passez en tant que paramètre lorsque vous créez par programmation une session temporaire pour un rôle ou un utilisateur fédéré. Les autorisations d'une session sont une combinaison des politiques basées sur l'identité de l'entité IAM (utilisateur ou rôle) utilisée pour créer la session et les politiques de session. Les autorisations peuvent également provenir d'une politique basée sur les ressources. Un refus explicite dans l'une de ces politiques annule l'autorisation.

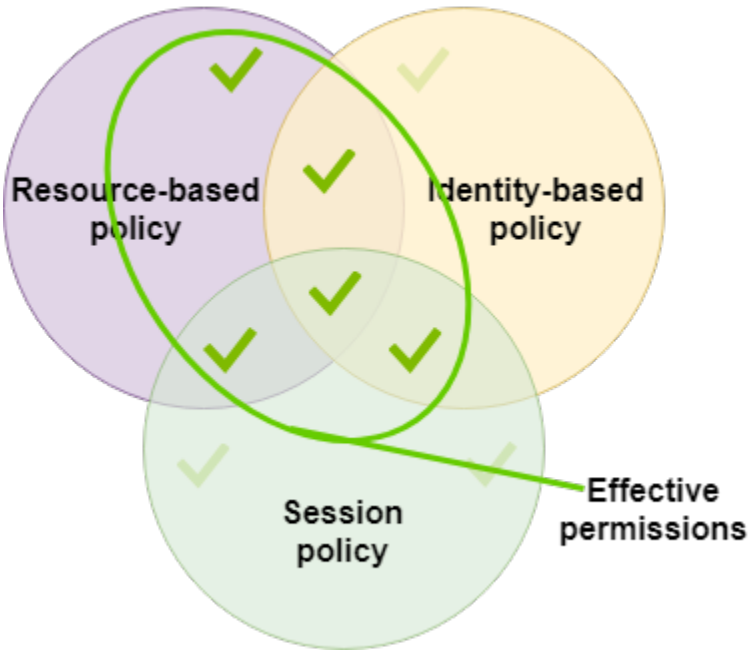
Vous pouvez créer une session de rôle et transmettre des politiques de session par programmation à l'aide des opérations d'API `AssumeRole`, `AssumeRoleWithSAML` ou `AssumeRoleWithWebIdentity`. Vous pouvez transmettre un seul document de politique de session en ligne JSON à l'aide du paramètre `Policy`. Vous pouvez utiliser le paramètre `PolicyArns` pour spécifier jusqu'à 10 politiques de session gérées. Pour plus d'informations sur la création d'une session de rôle, consultez Demande d'informations d'identification temporaires de sécurité.

Lorsque vous créez une session d'utilisateur fédéré, vous utilisez les clés d'accès de l'utilisateur IAM pour appeler par programmation l'opération d'API `GetFederationToken`. Vous devez également transmettre des politiques de session. Les autorisations de session obtenues sont une combinaison de la politique basée sur l'identité et de la politique de session. Pour plus d'informations sur la création d'une session d'utilisateur fédéré, consultez GetFederationToken : fédération via un broker d'identité personnalisé.

Une politique basée sur les ressources peut spécifier l'ARN de l'utilisateur ou du rôle en tant que principal. Dans ce cas, les autorisations de la politique basée sur les ressources sont ajoutées à la politique basée sur l'identité du rôle ou l'utilisateur avant la création de la session. La politique de session limite les autorisations totales accordées par la politique basée sur les ressources et la politique basée sur l'identité. Les autorisations de session résultantes sont l'intersection des politiques de session et des politiques basées sur les ressources, plus l'intersection des politiques de session et des politiques basées sur l'identité.

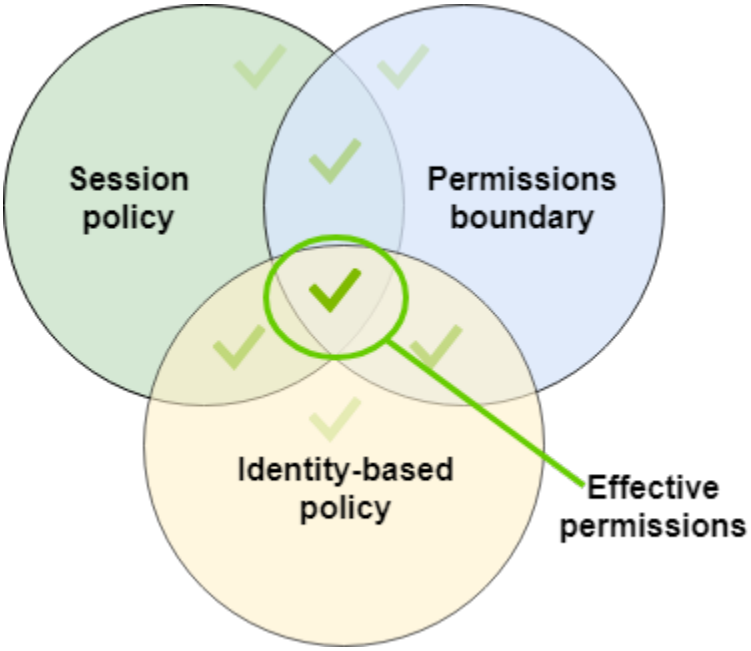


Une politique basée sur les ressources peut spécifier l'ARN de la session en tant que principal. Dans ce cas, les autorisations de la politique basée sur les ressources sont ajoutées après la création de la session. Les autorisations de politique basée sur les ressources ne sont pas limitées par la politique de session. La session résultante dispose de toutes les autorisations de la politique basée sur les ressources en plus de la combinaison de la politique basée sur l'identité et de la politique de session.



Une limite d'autorisations peut définir la limite maximale des autorisations pour un utilisateur ou un rôle qui est utilisé pour créer une session. Dans ce cas, les autorisations de session obtenues

sont une combinaison de la politique de session, de la limite d'autorisations et de la politique basée sur l'identité. Toutefois, une limite d'autorisations ne limite pas les autorisations accordées par une politique basée sur les ressources, qui spécifie l'ARN de la session résultante.



Politiques et utilisateur racine

Elle Utilisateur racine d'un compte AWS est affectée par certains types de politiques, mais pas par d'autres. Vous ne pouvez pas attacher des politiques basées sur l'identité à l'utilisateur racine, et vous ne pouvez pas définir la limite d'autorisations pour l'utilisateur racine. Pourtant, vous pouvez spécifier l'utilisateur racine en tant que principal dans une politique basée sur les ressources ou une liste de contrôle d'accès. Un utilisateur racine est toujours le membre d'un compte. Si ce compte est membre d'une organisation en AWS Organizations, l'utilisateur root est affecté par les SCP associés au compte.

Présentation des politiques JSON

La plupart des politiques sont stockées AWS sous forme de documents JSON. Les politiques basées sur l'identité et les politiques utilisées pour définir des autorisations sont des documents de politique JSON que vous attachez à un utilisateur ou à un rôle. Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Les SCP sont des documents de politique JSON à syntaxe restreinte que vous attachez à une unité AWS Organizations organisationnelle (UO). Les listes de contrôle d'accès sont également attachées à une ressource, mais vous devez utiliser une syntaxe différente. Les politiques de session sont des politiques JSON que vous fournissez lorsque vous endossez une session de rôle ou d'utilisateur fédéré.

Il n'est pas nécessaire pour vous de comprendre la syntaxe JSON. Vous pouvez utiliser l'éditeur visuel du AWS Management Console pour créer et modifier des politiques gérées par le client sans jamais utiliser JSON. Toutefois, si vous utilisez des politiques en ligne pour des groupes ou des politiques complexes, vous devez quand même créer et modifier ces politiques dans l'éditeur JSON à l'aide de la console. Pour plus d'informations sur l'utilisation de l'éditeur visuel, consultez Création de politiques IAM et Modification de politiques IAM.

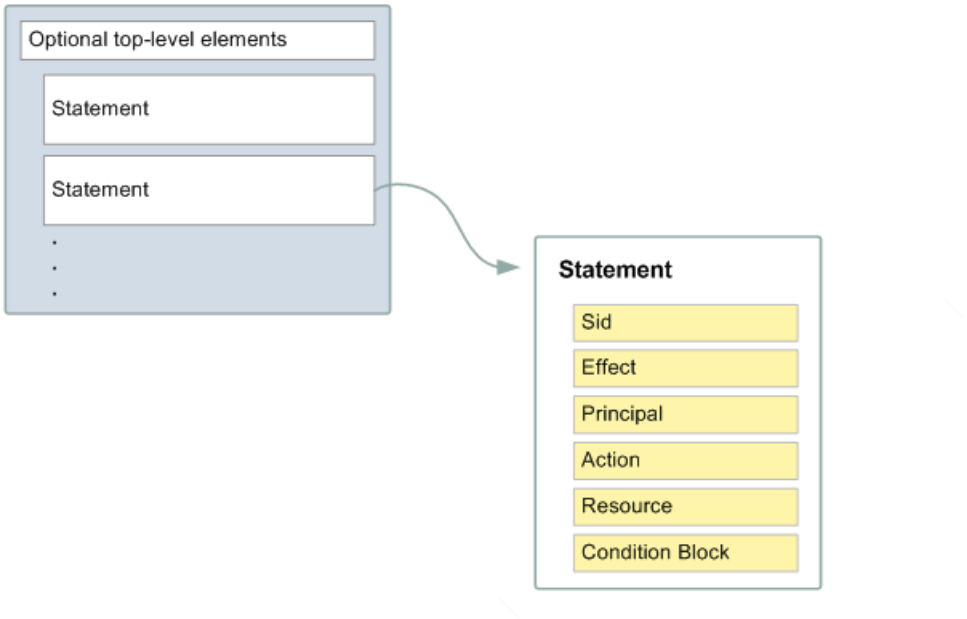
Lorsque vous créez ou modifiez une politique JSON, IAM peut effectuer une validation de politique pour vous aider à créer une politique efficace. IAM identifie les erreurs de syntaxe JSON, tandis que IAM Access Analyzer fournit des vérifications de politique supplémentaires avec des recommandations pour vous aider à affiner vos politiques. Pour en savoir plus sur la validation de politiques, veuillez consulter Validation de politiques IAM. Pour en savoir plus sur les vérifications des politiques IAM Access Analyzer et les recommandations exploitables, veuillez consulter Validation de politique IAM Access Analyzer.

Structure d'un document de politique JSON

Comme illustré dans la figure suivante, un document de politique JSON inclut les éléments suivants :

- Informations facultatives sur l'ensemble de la politique (en haut du document)
- Une ou plusieurs instructions individuelles

Chaque instruction inclut des informations sur une seule autorisation. Si une politique inclut plusieurs instructions, AWS applique une logique `OR` aux instructions lors de leur évaluation. Si plusieurs politiques s'appliquent à une demande, AWS applique une logique `OR` à toutes ces politiques lors de leur évaluation.



Les informations contenues dans une instruction sont situées dans une série d'éléments.

- Version : spécifiez la version du langage de politique que vous souhaitez utiliser. Nous vous recommandons de choisir la dernière version `2012-10-17`. Pour plus d'informations, consultez Éléments de politique JSON IAM : Version.
- Statement : utilisez cet élément de politique principal comme conteneur pour les éléments suivants. Vous pouvez inclure plus d'une instruction dans une politique.
- Sid (Facultatif) : incluez un ID d'instruction facultatif pour différencier vos instructions.
- Effect : utilisez `Allow` ou `Deny` pour indiquer si la politique autorise ou refuse l'accès.
- Principal (Obligatoire dans certaines circonstances uniquement) : si vous créez une politique basée sur les ressources, vous devez indiquer le compte, l'utilisateur, le rôle ou l'utilisateur fédéré auquel vous souhaitez autoriser ou refuser l'accès. Si vous créez une politique d'autorisations IAM à attacher à un utilisateur ou un rôle, vous ne pouvez pas inclure cet élément. Le principal est implicitement cet utilisateur ou rôle.
- Action : incluez la liste des actions autorisées ou refusées par la politique.
- Resource (Obligatoire dans certaines circonstances uniquement) : si vous créez une politique d'autorisations IAM, vous devez spécifier la liste des ressources auxquelles les actions s'appliquent. Si vous créez une politique basée sur les ressources, cet élément est facultatif. Si vous n'incluez pas cet élément, la ressource à laquelle l'action s'applique est la ressource à laquelle la politique est attachée.

- Condition (Facultatif) : spécifiez les circonstances dans lesquelles la politique accorde une autorisation.

Pour en savoir plus sur ces éléments et d'autres éléments de politique plus avancés, consultez Références des éléments de politique JSON IAM.

Plusieurs instructions et plusieurs politiques

Si vous souhaitez définir plusieurs autorisations pour une entité (utilisateur ou rôle), vous pouvez utiliser plusieurs instructions dans une seule politique. Vous pouvez également attacher plusieurs politiques. Si vous essayez de définir plusieurs autorisations dans une seule instruction, votre politique peut ne pas accorder l'accès que vous attendez. Nous vous recommandons de répartir les politiques par type de ressource.

En raison de la taille limitée des politiques, il peut être nécessaire d'en utiliser plusieurs pour les autorisations les plus complexes. Il est également conseillé de créer des regroupements fonctionnels d'autorisations dans une politique gérée par le client séparée. Par exemple, créez une politique pour la gestion des utilisateurs IAM, une pour la gestion automatique et une autre pour la gestion de compartiment S3. Quelle que soit la combinaison de plusieurs déclarations et de plusieurs politiques, AWS évalue vos politiques de la même manière.

Par exemple, la politique suivante comporte trois instructions, chacune définissant un ensemble séparé d'autorisations dans un seul compte. Les instructions définissent les opérations suivantes :

- La première instruction, avec un `Sid` (ID d'instruction) de `FirstStatement`, permet à l'utilisateur disposant de la politique attachée de modifier son propre mot de passe. L'élément `Resource` de cette instruction a pour valeur « * » (ce qui signifie « toutes les ressources »). Toutefois, dans la pratique, l'opération d'API `ChangePassword` (ou la commande CLI `change-password` équivalente) affecte uniquement le mot de passe de l'utilisateur qui fait la demande.
- La deuxième instruction permet à l'utilisateur de répertorier tous les compartiments Amazon S3 de son Compte AWS. L'élément `Resource` de cette instruction a pour valeur `"*"` (ce qui signifie « toutes les ressources »). Cependant, du fait que les politiques n'accordent pas l'accès aux ressources des autres comptes, l'utilisateur ne peut répertorier que les compartiments de son propre Compte AWS.
- La troisième instruction permet à l'utilisateur de répertorier les objets situés dans un compartiment appelé `confidential-data`, mais uniquement lorsque l'utilisateur est authentifié avec une authentification multi-facteur (MFA). L'élément `Condition` de la politique applique l'authentification MFA.

Lorsqu'une instruction de politique contient un élément `Condition`, l'instruction est effective uniquement lorsque l'élément `Condition` est true. Dans ce cas, la `Condition` est true lorsque l'utilisateur est authentifié MFA. Si l'utilisateur n'est pas authentifié MFA, cette `Condition` est false. Dans ce cas, la troisième instruction de cette politique ne s'applique pas et l'utilisateur n'a pas accès au compartiment `confidential-data`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "FirstStatement",
      "Effect": "Allow",
      "Action": ["iam:ChangePassword"],
      "Resource": "*"
    },
    {
      "Sid": "SecondStatement",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Sid": "ThirdStatement",
      "Effect": "Allow",
      "Action": [
        "s3:List*",
        "s3:Get*"
      ],
      "Resource": [
        "arn:aws:s3:::confidential-data",
        "arn:aws:s3:::confidential-data/*"
      ],
      "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
    }
  ]
}
```

Exemples de syntaxe d'une politique JSON

La politique basée sur l'identité suivante permet au principal implicite de répertorier un seul compartiment Amazon S3 nommé `example_bucket` :

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

La politique basée sur les ressources suivante peut être attachée à un compartiment Amazon S3. La politique permet aux membres d'un groupe spécifique Compte AWS d'effectuer toutes les actions Amazon S3 dans le compartiment nommé`mybucket`. Elle permet à n'importe quelle action d'être exécutée sur un compartiment ou les objets qu'il contient. (Du fait que la politique accorde une approbation uniquement au compte, les utilisateurs individuels du compte doivent se voir accorder des autorisations concernant les actions Amazon S3 spécifiées.)

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "1",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::account-id:root"]},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }]
}
```

Pour afficher des exemples de politiques sur des scénarios courants, consultez Exemples de politiques basées sur l'identité IAM.

Accorder les privilèges les plus faibles possible

Lorsque vous créez des politiques IAM, suivez le conseil de sécurité standard du moindre privilège, principe selon lequel il ne faut accorder que les autorisations requises pour une seule tâche. Déterminez les actions que les utilisateurs (et les rôles) doivent effectuer et élaborez des politiques leur permettant de réaliser uniquement ces tâches.

Commencez avec un ensemble d'autorisations minimum et accordez-en d'autres si nécessaire. Cette méthode est plus sûre que de commencer avec des autorisations trop permissives et d'essayer de les restreindre plus tard.

Comme alternative au moindre privilège, vous pouvez utiliser des politiques gérées par AWS ou des politiques accompagnées d'autorisations de `*` par caractère générique, pour commencer avec les politiques. Veuillez prendre en compte le risque de sécurité constitué par l'octroi, à vos principaux, de davantage d'autorisations que nécessaire pour accomplir leur tâche. Contrôlez ces principaux afin de savoir quelles autorisations ils utilisent. Écrivez ensuite des politiques de moindre privilège.

IAM propose plusieurs options pour vous aider à affiner les autorisations que vous octroyez.

- Understand access level groupings (Comprendre les regroupements de niveau d'accès) : vous pouvez utiliser des regroupements de niveaux d'accès pour comprendre le niveau d'accès accordé par la politique. Les actions de politique sont classées en tant que `List`, `Read`, `Write`, `Permissions management` ou `Tagging`. Par exemple, vous pouvez choisir des actions dans les niveaux d'accès `List` et `Read` pour accorder un accès en lecture seule à vos utilisateurs. Pour savoir comment utiliser les récapitulatifs de politiques pour comprendre les autorisations de niveau d'accès, consultez Comprendre les niveaux d'accès dans les résumés des politiques.
- Valider vos politiques : vous pouvez effectuer une validation de politique à l'aide d'IAM Access Analyzer lorsque vous créez et modifiez des politiques JSON. Nous vous recommandons d'examiner et de valider toutes vos politiques existantes. IAM Access Analyzer fournit plus de 100 vérifications de politiques pour valider vos politiques. Il génère des avertissements de sécurité lorsqu'une instruction de votre politique autorise un accès que nous considérons comme trop permissif. Vous pouvez utiliser les recommandations exploitables contenues dans les avertissements de sécurité lorsque vous travaillez à l'octroi du moindre privilège. Pour en savoir plus sur les vérifications de politique fournies par IAM Access Analyzer, veuillez consulter IAM Access Analyzer policy validation (Validation de politique IAM Access Analyzer).
- Générer une politique basée sur l'activité d'accès : pour affiner les autorisations que vous octroyez, vous pouvez générer une politique IAM basée sur l'activité d'accès pour une entité IAM (utilisateur ou rôle). IAM Access Analyzer examine vos AWS CloudTrail journaux et génère un modèle de

politique contenant les autorisations utilisées par l'entité au cours de la période spécifiée. Vous pouvez utiliser le modèle pour créer une politique gérée avec des autorisations affinées, puis l'attacher à l'entité IAM. Ainsi, vous accordez uniquement les autorisations dont l'utilisateur ou le rôle a besoin pour interagir avec les AWS ressources correspondant à votre cas d'utilisation spécifique. Pour en savoir plus, veuillez consulter la section Générer des politiques basées sur l'activité d'accès.

- Utiliser les dernières informations consultées : les dernières informations consultées sont une autre fonction pouvant vous aider avec le principe de moindre privilège. Consultez ces informations sous l'onglet Access Advisor sur la page des détails de la console IAM pour un utilisateur, un groupe, un rôle ou une politique IAM. Les dernières informations consultées incluent également des informations sur les dernières actions consultées pour certains services, tels qu'Amazon EC2, IAM, Lambda et Amazon S3. Si vous vous connectez à l'aide AWS Organizations des informations d'identification du compte de gestion, vous pouvez consulter les informations du dernier accès au service dans la AWS Organizationssection de la console IAM. Vous pouvez également utiliser l' AWS API AWS CLI or pour récupérer un rapport sur les dernières informations consultées pour les entités ou les politiques dans IAM ou Organizations. Elles vous permettent d'identifier toute autorisation inutile. Vous pouvez ainsi peaufiner vos politiques IAM ou Organizations afin qu'elles respectent au plus près le principe du moindre privilège. Pour plus d’informations, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.
- Passez en revue les événements du compte dans AWS CloudTrail — Pour réduire davantage les autorisations, vous pouvez consulter les événements de votre compte dans l'historique des AWS CloudTrail événements. CloudTrail les journaux d'événements contiennent des informations détaillées sur les événements que vous pouvez utiliser pour réduire les autorisations prévues par la politique. Les journaux incluent uniquement les actions et les ressources dont vos entités IAM ont besoin. Pour plus d'informations, consultez la section Affichage CloudTrail des événements dans la CloudTrail console dans le guide de AWS CloudTrail l'utilisateur.

Pour en savoir plus, consultez les rubriques de politiques pour des services individuels qui fournissent des exemples de rédaction de politiques pour des ressources spécifiques à des services.

- Authentification et contrôle d'accès pour Amazon DynamoDB dans le Manuel du développeur Amazon DynamoDB
- Utilisation de politiques de compartiment et de politiques d'utilisateur dans le guide de l'utilisateur du service de stockage simple Amazon

- Présentation de la liste de contrôle d'accès (ACL) dans le guide de l'utilisateur service de stockage simple Amazon

Politiques gérées et politiques en ligne

Lorsque vous définissez les autorisations pour une identité dans IAM, vous devez décider si vous souhaitez utiliser une politique gérée par AWS , une politique gérée par le client ou une politique en ligne. Les rubriques suivantes contiennent d'autres informations sur chaque type de politique basée sur l'identité et sur la façon de les utiliser.

Rubriques

- AWS politiques gérées
- Politiques gérées par le client
- Politiques en ligne
- Choix entre les politiques gérées et les politiques en ligne
- Premiers pas avec les politiques gérées
- Transformer une politique en ligne en politique gérée
- Politiques gérées déconseillées AWS

AWS politiques gérées

Une politique gérée par AWS est une politique autonome qui est créée et gérée par AWS. Ces politiques sont dites autonomes, ce qui signifie que chaque politique a son propre Amazon Resource Name (ARN) incluant le nom de la politique. Par exemple, il `arn:aws:iam::aws:policy/IAMReadOnlyAccess` s'agit d'une politique AWS gérée. Pour plus d'informations sur les ARN, consultez ARN IAM. Pour obtenir la liste des politiques AWS gérées pour Services AWS, consultez la section stratégies AWS gérées.

AWS les politiques gérées vous permettent d'attribuer facilement les autorisations appropriées aux utilisateurs, aux groupes et aux rôles. Plus rapide que d'écrire vous-même les politiques, cela inclut des autorisations pour de nombreux cas d'utilisation courants.

Vous ne pouvez pas modifier les autorisations définies dans les politiques AWS gérées. AWS met à jour de temps en temps les autorisations définies dans une politique AWS gérée. Dans ce cas AWS , la mise à jour affecte toutes les entités principales (utilisateurs, groupes et rôles) auxquelles la politique est attachée. AWS est le plus susceptible de mettre à jour une politique AWS gérée

lorsqu'un nouveau AWS service est lancé ou que de nouveaux appels d'API sont disponibles pour les services existants. Par exemple, la politique AWS gérée appelée ReadOnlyAccessfournit un accès en lecture seule à tous les AWS services et ressources. Lors du AWS lancement d'un nouveau service, AWS met à jour la ReadOnlyAccesspolitique pour ajouter des autorisations en lecture seule pour le nouveau service. Les autorisations mises à jour s'appliquent à toutes les entités du principal auxquelles la politique est attachée.

Les politiques de AWS gestion de l'accès complet définissent les autorisations pour les administrateurs de services en accordant un accès complet à un service.

- AmazonDynamoDB FullAccess
- IAM FullAccess

Les politiques AWS gérées par les utilisateurs expérimentés fournissent un accès complet aux AWS services et aux ressources, mais ne permettent pas de gérer les utilisateurs et les groupes.

- AWSCodeCommitPowerUser
- AWSKeyManagementServicePowerUser

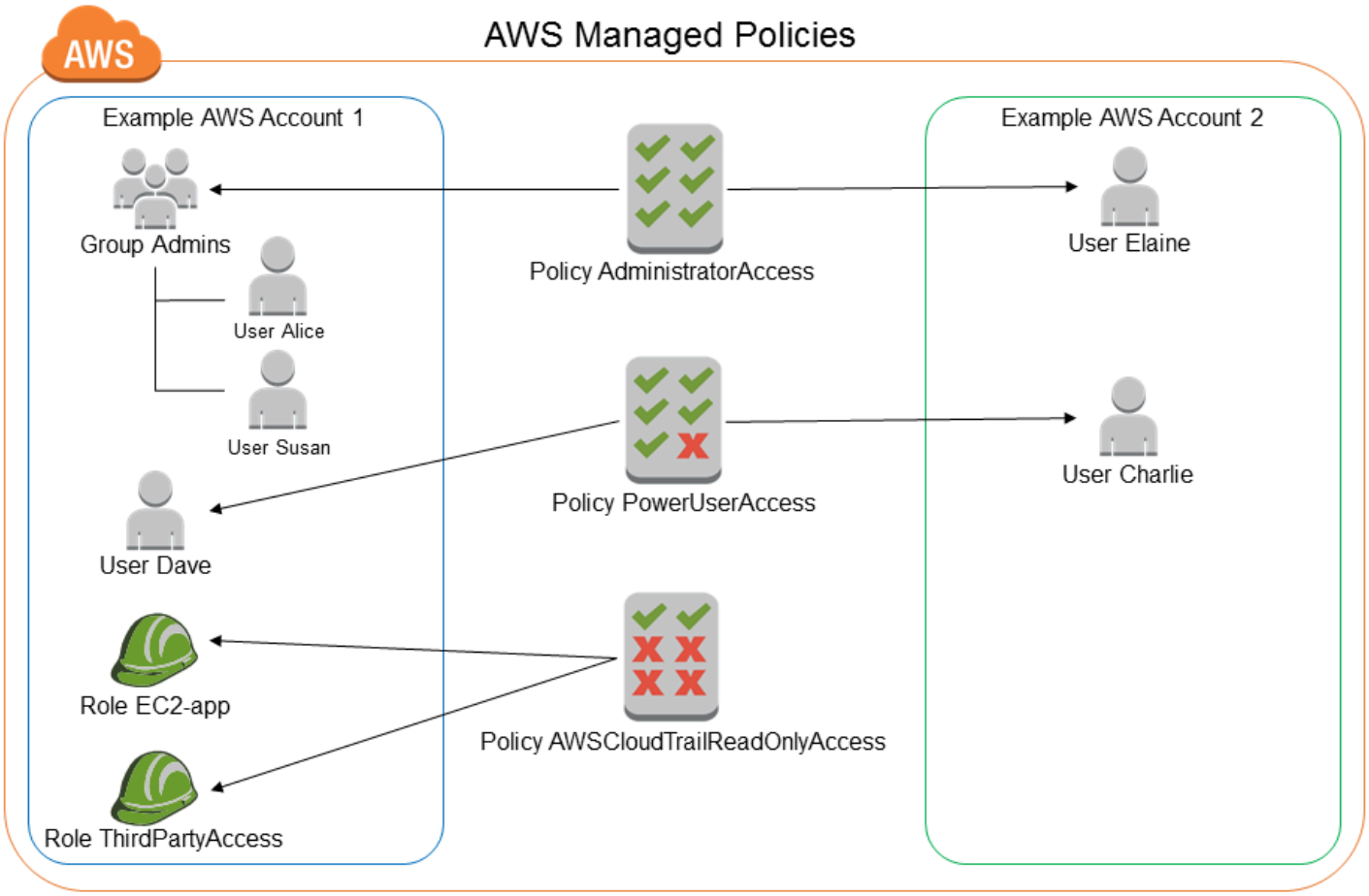
Les politiques de AWS gestion de l'accès partiel fournissent des niveaux d'accès spécifiques aux AWS services sans autoriser les autorisations de niveau d'accès liées à la gestion des autorisations.

- AmazonMobileAnalyticsWriteOnlyAccess
- Amazon EC2 ReadOnlyAccess

Une catégorie particulièrement utile de politiques AWS gérées est celle conçue pour les fonctions professionnelles. Ces politiques correspondent à des fonctions professionnelles couramment utilisées dans le secteur de l'informatique et facilitent l'octroi d'autorisations pour ces fonctions professionnelles. L'un des principaux avantages de l'utilisation des politiques relatives aux fonctions de travail est qu'elles sont maintenues et mises à jour à AWS mesure que de nouveaux services et opérations d'API sont introduits. Par exemple, la fonction AdministratorAccessjob fournit un accès complet et une délégation d'autorisations à chaque service et ressource qu'il contient AWS. Nous vous recommandons de n'utiliser cette politique que pour l'administrateur de compte. Pour les utilisateurs expérimentés qui ont besoin d'un accès complet à tous les services, à l'exception de l'accès limité à IAM et aux Organizations, utilisez la fonction PowerUserAccessjob. Pour obtenir la

liste et la description des politiques de fonctions professionnelles, consultez la page AWS politiques gérées pour les fonctions professionnelles.

Le schéma suivant illustre les politiques AWS gérées. Le diagramme montre trois politiques AWS gérées : AdministratorAccessPowerUserAccess, et AWS CloudTrailReadOnlyAccess. Notez qu'une seule politique AWS gérée peut être attachée à des entités principales dans différentes entités principales Comptes AWS, et à différentes entités principales dans une seule Compte AWS.



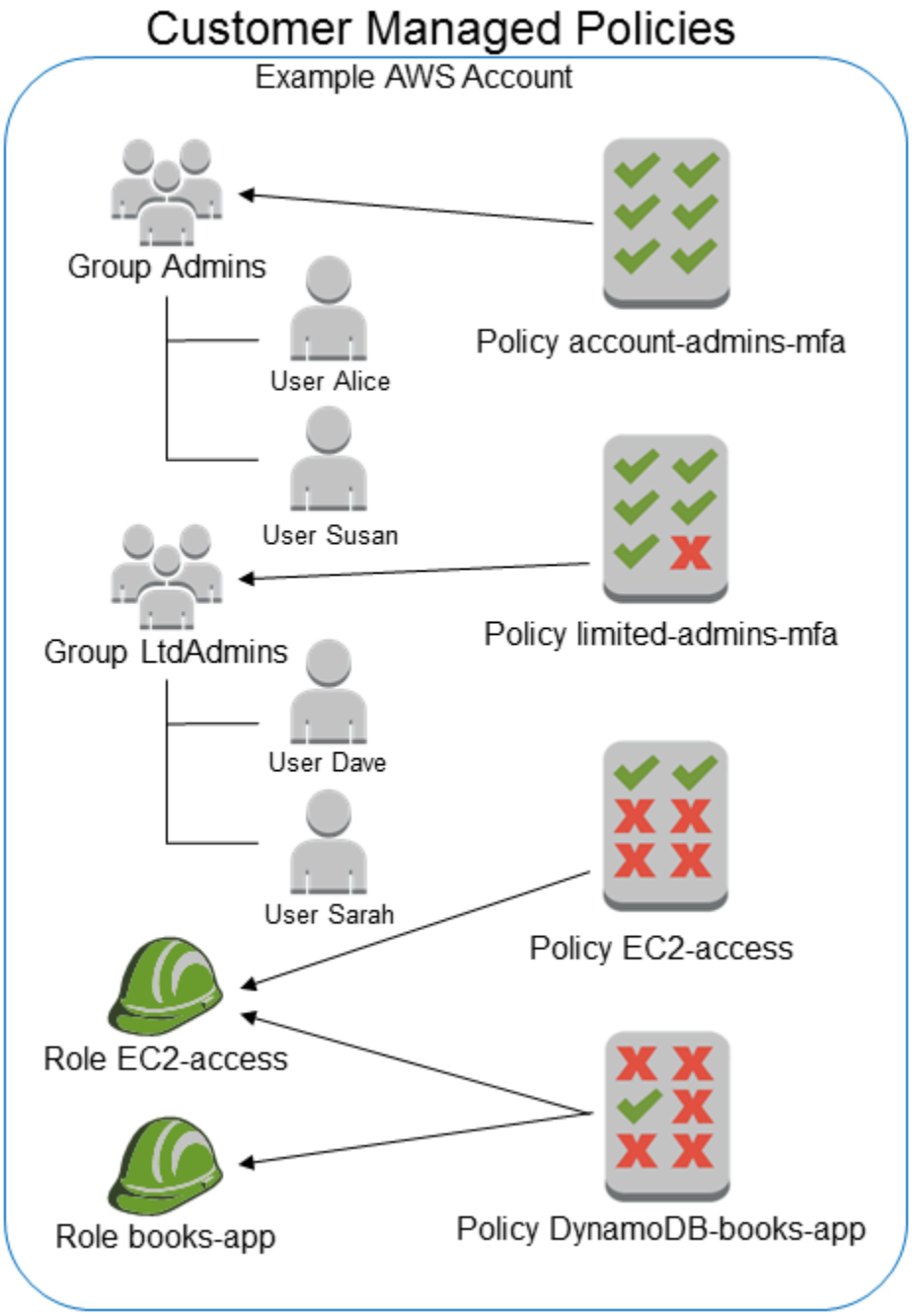
Politiques gérées par le client

Vous pouvez créer vos propres politiques autonomes Compte AWS que vous pouvez associer aux entités principales (utilisateurs, groupes et rôles). Vous créez ces politiques gérées par le client pour vos cas d'utilisation spécifiques, et vous pouvez les modifier et les mettre à jour aussi souvent que vous le souhaitez. À l'instar des politiques AWS gérées, lorsque vous attachez une politique à une entité principale, vous accordez à l'entité les autorisations définies dans la stratégie. Lorsque vous mettez à jour les autorisations dans la politique, les changements sont appliqués à toutes les entités du principal auxquelles la politique est attachée.

Pour créer une politique gérée par le client, une bonne méthode consiste à commencer par copier une politique gérée par AWS . De cette façon, vous êtes sûr que la politique est correcte au départ ; il vous suffit ensuite de la personnaliser en fonction de votre environnement.

Le diagramme suivant illustre des politiques gérées par le client. Chaque politique est une entité dans IAM avec son propre Amazon Resource Name (ARN) dans lequel figure le nom de la politique. Notez qu'une même politique peut être attachée à plusieurs entités du principal. Par exemple, la politique DynamoDB-books-app est attachée à deux rôles IAM différents.

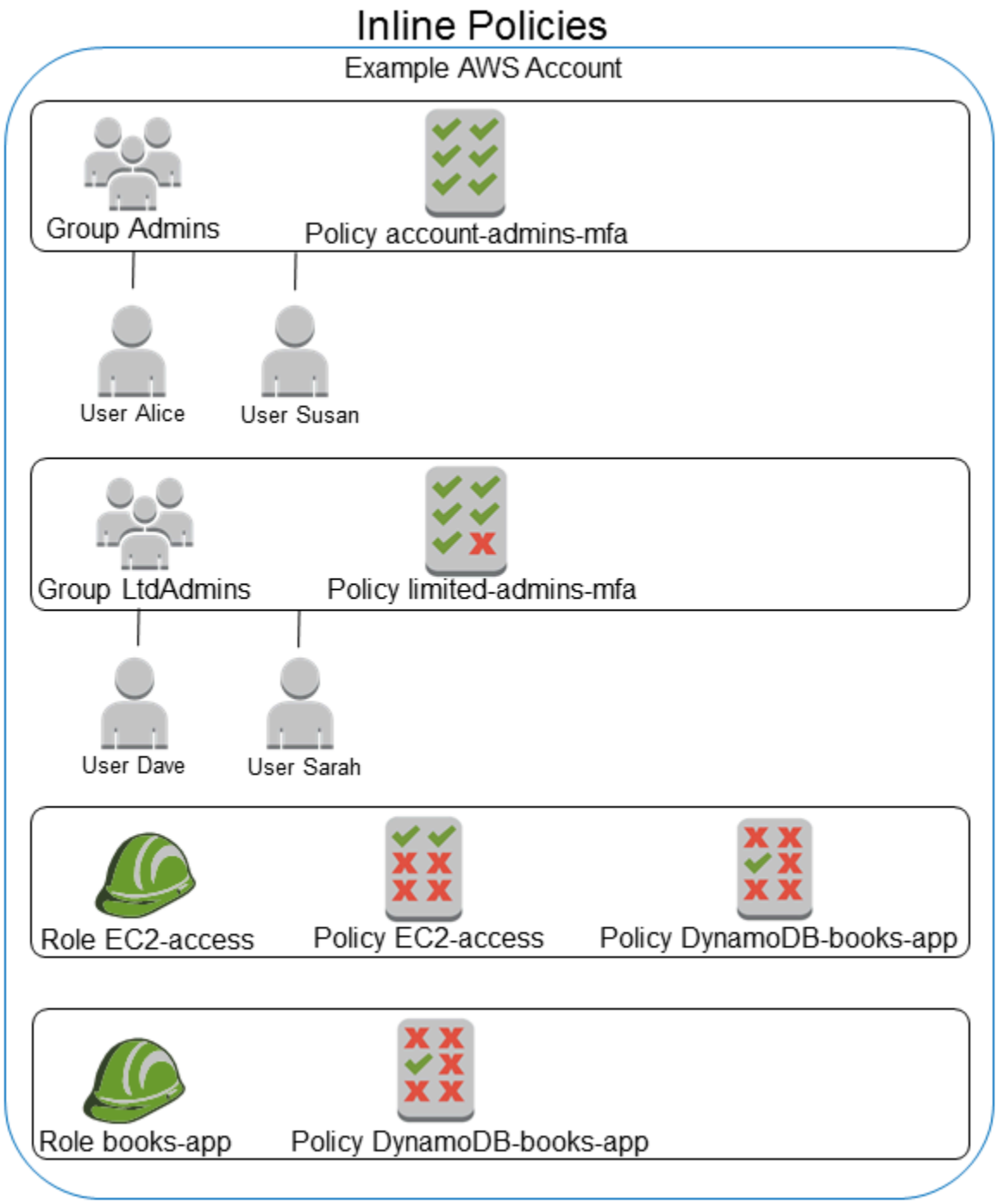
Pour plus d’informations, consultez Création de politiques IAM.



Politiques en ligne

Une politique en ligne est une politique créée pour une identité IAM (utilisateur, groupe ou rôle). Les politiques intégrées maintiennent une one-to-one relation stricte entre une politique et une identité. Elles sont supprimées lorsque vous supprimez l'identité. Vous pouvez créer une politique et l'intégrer dans une identité, soit lors de la création de l'identité, soit ultérieurement. Si une politique peut s'appliquer à plusieurs entités, il est préférable d'utiliser une politique gérée.

Le diagramme suivant illustre des politiques en ligne. Chaque politique fait partie intégrante de l'utilisateur, du groupe ou du rôle. Notez que deux rôles incluent la même politique (politique DynamoDB-books-app), sans toutefois partager une même politique. Chaque rôle possède sa propre copie de la politique.



Choix entre les politiques gérées et les politiques en ligne

Prenez en compte vos cas d'utilisation lorsque vous devez choisir entre les politiques gérées et les politiques en ligne. Dans la plupart des cas, nous vous recommandons d'utiliser des politiques gérées plutôt que des politiques en ligne.

> **Note**
>
> Vous pouvez utiliser à la fois des politiques gérées et des politiques en ligne pour définir des autorisations communes et uniques pour une entité du principal.

Les politiques gérées fournissent les fonctions suivantes :

Réutilisation

Une même politique gérée peut être attachée à plusieurs entités du principal (utilisateurs, groupes et rôles). Vous pouvez créer une bibliothèque de politiques qui définissent des autorisations utiles pour vous Compte AWS, puis associer ces politiques aux entités principales selon vos besoins.

Gestion centralisée des modifications

Lorsque vous modifiez une politique gérée, le changement est appliqué à toutes les entités du principal auxquelles la politique est attachée. Par exemple, si vous souhaitez ajouter une autorisation pour une nouvelle AWS API, vous pouvez mettre à jour une politique gérée par le client ou associer une politique AWS gérée pour ajouter l'autorisation. Si vous utilisez une politique AWS gérée, AWS mettez-la à jour. Lorsqu'une stratégie gérée est mise à jour, les modifications sont appliquées à toutes les entités principales auxquelles la stratégie gérée est attachée. En revanche, pour modifier une politique intégrée, vous devez modifier individuellement chaque identité contenant la politique intégrée. Par exemple, si un groupe et un rôle contiennent tous les deux la même politique en ligne, vous devez modifier individuellement les deux entités du principal pour modifier la politique.

Versioning et restauration

Lorsque vous modifiez une politique gérée par le client, la politique modifiée ne remplace pas la politique existante. À la place, IAM crée une nouvelle version de la politique gérée. IAM stocke jusqu'à cinq versions de vos politiques gérées par le client. Vous pouvez utiliser les versions de politique pour restaurer une version antérieure, si nécessaire.

> **Note**
>
> Une version de politique est différente d'un élément de politique `Version`. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. Pour en savoir plus sur les versions de politiques, consultez the section called “Gestion des versions des politiques IAM”. Pour en savoir plus sur l'élément de politique `Version`, consultez Éléments de politique JSON IAM : Version.

Délégation de la gestion des autorisations

Vous pouvez autoriser les utilisateurs de votre site Compte AWS à joindre et à détacher des politiques tout en gardant le contrôle sur les autorisations définies dans ces politiques. Pour ce faire, vous pouvez ainsi désigner certains utilisateurs en tant qu'administrateurs disposant de droits complets, autrement dit autorisés à créer, mettre à jour et supprimer des politiques. D'autres utilisateurs peuvent ensuite être désignés en tant qu'administrateurs limitées. Ces administrateurs restreints pouvent attacher des politiques à d'autres entités du principal, mais uniquement celles que vous leur avez autorisés à attacher.

Pour plus d'informations sur la délégation de la gestion des autorisations, consultez Contrôle de l'accès aux politiques.

Limites de caractères des politiques plus grandes

La taille maximale de caractères des politiques gérées est supérieure à la taille maximale de caractères des politiques en ligne. Si vous atteignez la limite de taille de caractères pour la politique en ligne, vous pouvez créer d'autres groupes IAM et associer la politique gérée au groupe.

Pour plus d'informations sur les quotas et les limites, consultez IAM et quotas AWS STS.

Mises à jour automatiques pour les politiques AWS gérées

AWS gère les politiques AWS gérées et les met à jour lorsque cela est nécessaire, par exemple pour ajouter des autorisations pour de nouveaux AWS services, sans que vous ayez à apporter de modifications. Les mises à jour sont automatiquement appliquées aux principales entités auxquelles vous avez attaché la politique AWS gérée.

Utilisation de politiques en ligne

Les politiques intégrées sont utiles si vous souhaitez maintenir une one-to-one relation stricte entre une stratégie et l'identité à laquelle elle est appliquée. Par exemple, si vous voulez éviter que les autorisations d'une politique ne soient accidentellement affectées à une identité autre que celle à laquelle elles sont destinées. Lorsque vous utilisez une politique en ligne, ses autorisations ne peuvent pas être attachées par inadvertance à la mauvaise identité. En outre, lorsque vous utilisez le AWS Management Console pour supprimer cette identité, les politiques intégrées à l'identité sont également supprimées car elles font partie de l'entité principale.

Premiers pas avec les politiques gérées

Nous vous recommandons d'utiliser des politiques qui accordent le moins de privilèges ou d'accorder uniquement les autorisations requises pour effectuer une tâche. Le moyen le plus sûr d'octroyer le moindre privilège consiste à écrire une politique gérée par le client contenant uniquement les autorisations requises par votre équipe. Vous devez créer un processus pour autoriser votre équipe à demander plus d'autorisations si nécessaire. Il faut du temps et de l’expertise pour créer des politiques gérées par le client IAM qui ne fournissent à votre équipe que les autorisations dont elle a besoin.

Pour commencer à ajouter des autorisations à vos identités IAM (utilisateurs, groupes d'utilisateurs et rôles), vous pouvez utiliserAWS politiques gérées. AWS les politiques gérées n'accordent pas les autorisations du moindre privilège. Vous devez prendre en compte le risque de sécurité constitué par l'octroi, à vos principaux, de davantage d'autorisations que nécessaire pour accomplir leur tâche.

Vous pouvez associer des politiques AWS gérées, y compris des fonctions de travail, à n'importe quelle identité IAM. Pour plus d’informations, consultez Ajout et suppression d'autorisations basées sur l'identité IAM.

Pour passer aux autorisations du moindre privilège, vous pouvez exécuter AWS Identity and Access Management Access Analyzer pour surveiller les principaux à l'aide de politiques AWS gérées. Lorsque vous savez quelles autorisations ils utilisent, vous pouvez écrire ou générer une politique gérée par le client avec uniquement les autorisations requises pour votre équipe. Cela est moins sûr, mais offre plus de flexibilité à mesure que vous apprenez comment votre équipe les utilise AWS. Pour plus d’informations, consultez Génération d'une politique IAM Access Analyzer.

AWS les politiques gérées sont conçues pour fournir des autorisations pour de nombreux cas d'utilisation courants. Pour plus d'informations sur les politiques AWS gérées conçues pour des fonctions professionnelles spécifiques, consultezAWS politiques gérées pour les fonctions professionnelles.

Pour obtenir la liste des politiques AWS gérées, consultez le Guide de référence des politiques AWS gérées.

Transformer une politique en ligne en politique gérée

Si vous avez des politiques en ligne dans votre compte, vous pouvez les convertir en politiques gérées. Pour ce faire, copiez la politique dans une nouvelle politique gérée. Ensuite, attachez la nouvelle politique à l'identité qui inclut la politique en ligne. Ensuite, supprimez la politique en ligne.

Pour convertir une politique en ligne en politique gérée

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), Users (Utilisateurs) ou Roles (Rôles).
3. Choisissez dans la liste le nom du groupe d'utilisateurs, de l'utilisateur ou du rôle pour lequel vous souhaitez supprimer la politique.
4. Choisissez l'onglet Permissions (Autorisations).
5. Pour les groupes d'utilisateurs, sélectionnez le nom de la politique en ligne que vous voulez supprimer. Pour les utilisateurs et les rôles, choisissez Afficher ***n*** autres, si nécessaire, puis développez la politique en ligne que vous voulez supprimer.
6. Choisissez Copier pour copier le document de politique JSON pour la politique.
7. Dans le panneau de navigation, choisissez Politiques.
8. Choisissez Créer une politique, puis l'option JSON.
9. Remplacez le texte existant par votre texte de politique JSON, puis choisissez Suivant.
10. Saisissez un nom et une description facultative pour votre politique et choisissez Créer une politique.
11. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), Users (Utilisateurs) ou Roles (Rôles), puis sélectionnez le nom du groupe d'utilisateurs, de l'utilisateur ou du rôle pour lequel vous souhaitez supprimer la politique.
12. Choisissez l'onglet Autorisations, puis Ajouter des autorisations.
13. Pour les groupes, cochez la case à côté du nom de votre nouvelle politique. Sélectionnez Add permissions (Ajouter des autorisations), puis Attach policy (Attacher une politique). Pour les utilisateurs ou les rôles, choisissez Ajouter des autorisations. Sur la page suivante, choisissez

Attacher directement les politiques existantes, cochez la case en regard du nom de votre nouvelle politique, choisissez Suivant, puis Ajouter des autorisations.

Vous êtes renvoyé à la page Summary (Résumé) de votre groupe d'utilisateurs, utilisateur ou rôle.

14. Cochez la case en regard de la politique en ligne que vous voulez supprimer et choisissez Supprimer.

Politiques gérées déconseillées AWS

Pour simplifier l'attribution des autorisations, AWS fournit des politiques gérées, c'est-à-dire des politiques prédéfinies prêtes à être associées à vos utilisateurs, groupes et rôles IAM.

Il est parfois AWS nécessaire d'ajouter une nouvelle autorisation à une politique existante, par exemple lorsqu'un nouveau service est introduit. L'ajout d'une nouvelle autorisation à une politique existante n'impacte aucune des fonctions, pas plus qu'elle ne les supprime.

Toutefois, AWS vous pouvez choisir de créer une nouvelle politique lorsque les modifications nécessaires pourraient avoir un impact sur les clients si elles étaient appliquées à une politique existante. Par exemple, la suppression d'autorisations d'une politique existante peut affecter les autorisations d'une entité ou application IAM qui en dépend, ce qui risque de perturber une opération critique.

Par conséquent, lorsqu'une telle modification est requise, AWS crée une toute nouvelle politique avec les modifications requises et la met à la disposition des clients. L'ancienne politique est ensuite marquée comme étant obsolète. Une icône d'avertissement s'affiche en regard d'une politique gérée obsolète dans la liste Politiques de la console IAM.

Les caractéristiques d'une politique obsolète sont les suivantes :

- Elle continue à fonctionner pour tous les utilisateurs, groupes et rôles actuellement attachés. Aucun élément ne cesse de fonctionner.
- Elle ne peut pas être attachée à de nouveaux utilisateurs, groupes ou rôles. Si vous la détachez d'une entité actuelle, vous ne pouvez plus la rattacher par la suite.
- Une fois qu'elle a été détachée de toutes les entités actuelles, elle n'est plus visible et ne peut plus être utilisée.

Si elle est requise pour un utilisateur, groupe ou rôle, vous devez attacher la nouvelle politique. Lorsqu'une politique est signalée comme étant obsolète, nous vous recommandons de prendre immédiatement les mesures nécessaires pour attacher l'ensemble des utilisateurs, groupes et rôles à la politique de remplacement et les détacher de la politique obsolète. Pour éviter les risques liés à l'utilisation d'une politique obsolète, il convient d'utiliser la politique de remplacement.

Établissez des barrières de sécurité en matière d'autorisations à l'aide de périmètres de données

Les barrières de protection du périmètre des données sont conçues pour servir de limites permanentes afin de protéger vos données sur un large éventail de comptes et de AWS ressources. Les périmètres de données suivent les meilleures pratiques de sécurité IAM pour établir des barrières d'autorisation sur plusieurs comptes. Ces garde-fous relatifs aux autorisations à l'échelle de l'organisation ne remplacent pas vos contrôles d'accès précis existants. Ils fonctionnent plutôt comme des contrôles d'accès grossiers qui aident à améliorer votre stratégie de sécurité en garantissant que les utilisateurs, les rôles et les ressources respectent un ensemble de normes de sécurité définies.

Un périmètre de données est un ensemble de barrières d'autorisation dans votre AWS environnement qui permettent de garantir que seules vos identités fiables accèdent aux ressources fiables des réseaux attendus.

- Identités fiables : principaux (rôles ou utilisateurs IAM) de vos AWS comptes et AWS services agissant en votre nom.
- Ressources fiables : ressources détenues par vos AWS comptes ou par AWS des services agissant en votre nom.
- Réseaux attendus : vos centres de données sur site et vos clouds privés virtuels (VPC), ou réseaux de AWS services agissant en votre nom.

> **Note**
>
> Dans certains cas, vous devrez peut-être étendre votre périmètre de données pour inclure également l'accès de vos partenaires commerciaux de confiance. Vous devez prendre en compte tous les modèles d'accès aux données prévus lorsque vous créez une définition des identités fiables, des ressources fiables et des réseaux attendus spécifiques à votre entreprise et à l'utilisation que vous en faites Services AWS.

Les contrôles du périmètre des données doivent être traités comme tout autre contrôle de sécurité dans le cadre du programme de sécurité de l'information et de gestion des risques. Cela signifie que vous devez effectuer une analyse des menaces afin d'identifier les risques potentiels dans votre environnement cloud, puis, en fonction de vos propres critères d'acceptation des risques, sélectionner et mettre en œuvre des contrôles de périmètre de données appropriés. Pour mieux orienter l'approche itérative basée sur les risques en matière de mise en œuvre du périmètre des données, vous devez comprendre quels risques de sécurité et quels vecteurs de menace sont pris en compte par les contrôles du périmètre des données, ainsi que vos priorités en matière de sécurité.

Contrôles du périmètre des données

Les contrôles grossiers du périmètre des données vous aident à atteindre six objectifs de sécurité distincts sur trois périmètres de données grâce à la mise en œuvre de différentes combinaisons et clés de condition. Types de politique

Périmètre	Objectif de contrôle	Utilisation	Appliqué le	Clés contextuelles de condition globale
Identity	Seules les identités fiables peuvent accéder à mes ressources	Politique basée sur une ressource	Ressources	lois : PrincipalOrgID lois : PrincipalOrgPaths
	Seules les identités fiables sont autorisées sur mon réseau	Politique de point de terminaison d'un VPC	Réseau	lois : PrincipalAccount lois : PrincipalIsAwsService
Ressources	Vos identités ne peuvent accéder qu'à des ressources fiables	SCP	Identités	lois : ResourceOrgID lois : ResourceOrgPaths lois : ResourceAccount

Périmètre	Objectif de contrôle	Utilisation	Appliqué le	Clés contextuelles de condition globale
	Seules les ressources fiables sont accessibles depuis votre réseau	Politique de point de terminaison d'un VPC	Réseau	
Réseau	Vos identités ne peuvent accéder aux ressources qu'à partir des réseaux attendus	SCP	Identités	lois : SourceIp lois : SourceVpc lois : SourceVpce
	Vos ressources ne sont accessibles qu'à partir des réseaux attendus	Politique basée sur une ressource	Ressources	AWS : via AWSService lois : PrincipalIsAwsService

Vous pouvez penser que les périmètres de données créent une limite ferme autour de vos données afin d'empêcher les modèles d'accès involontaires. Bien que les périmètres de données puissent empêcher tout accès involontaire à grande échelle, vous devez tout de même prendre des décisions précises en matière de contrôle d'accès. L'établissement d'un périmètre de données ne diminue en rien la nécessité d'affiner en permanence les autorisations en utilisant des outils tels que IAM Access Analyzer dans le cadre de votre parcours vers le principe du moindre privilège.

Périmètre d'identité

Un périmètre d'identité est un ensemble de contrôles d'accès préventifs grossiers qui permettent de garantir que seules les identités fiables peuvent accéder à vos ressources et que seules les identités fiables sont autorisées sur votre réseau. Les identités fiables incluent les principaux (rôles ou utilisateurs) de vos AWS comptes et les AWS services agissant en votre nom. Toutes les autres

identités sont considérées comme non fiables et sont interdites par le périmètre d'identité à moins qu'une exception explicite ne soit accordée.

Les clés de condition globales suivantes permettent de renforcer les contrôles du périmètre d'identité. Utilisez ces clés dans les politiques basées sur les ressources pour restreindre l'accès aux ressources, ou dans les politiques de point de terminaison VPC pour restreindre l'accès à vos réseaux.

- lois : PrincipalOrg ID— Vous pouvez utiliser cette clé de condition pour vous assurer que les principaux IAM à l'origine de la demande appartiennent à l'organisation spécifiée dans. AWS Organizations
- aws : PrincipalOrg Chemins— Vous pouvez utiliser cette clé de condition pour vous assurer que l'utilisateur IAM, le rôle IAM, l'utilisateur fédéré ou le A Utilisateur racine d'un compte AWS qui fait la demande appartient à l'unité organisationnelle (UO) spécifiée dans. AWS Organizations
- lois : PrincipalAccount— Vous pouvez utiliser cette clé de condition pour garantir que les ressources ne sont accessibles qu'au compte principal que vous spécifiez dans la politique.
- lois : PrincipalIs AWSServiceet lois : SourceOrg ID (alternativement aws : SourceOrg Chemins etlois : SourceAccount) — Vous pouvez utiliser ces clés de condition pour vous assurer que lorsque Service AWS les principaux accèdent à vos ressources, ils le font uniquement pour le compte d'une ressource de l'organisation, de l'unité organisationnelle ou d'un compte dans lequel elles se trouvent. AWS Organizations

Pour plus d'informations, voir Établissement d'un périmètre de données sur AWS : Autoriser uniquement les identités fiables à accéder aux données de l'entreprise.

Périmètre des ressources

Un périmètre de ressources est un ensemble de contrôles d'accès préventifs grossiers qui permettent de garantir que vos identités ne peuvent accéder qu'aux ressources fiables et que seules les ressources fiables sont accessibles depuis votre réseau. Les ressources fiables incluent les ressources détenues par vos AWS comptes ou par AWS des services agissant en votre nom.

Les clés de condition globales suivantes permettent de renforcer les contrôles du périmètre des ressources. Utilisez ces clés dans les politiques de contrôle des services (SCP) pour restreindre les ressources auxquelles vos identités peuvent accéder, ou dans les politiques de point de terminaison des VPC pour restreindre les ressources accessibles depuis vos réseaux.

- lois : ResourceOrg ID— Vous pouvez utiliser cette clé de condition pour vous assurer que la ressource à laquelle vous accédez appartient à l'organisation spécifiée dans AWS Organizations.
- aws : ResourceOrg Chemins— Vous pouvez utiliser cette clé de condition pour vous assurer que la ressource à laquelle vous accédez appartient à l'unité organisationnelle (UO) spécifiée dans AWS Organizations.
- lois : ResourceAccount— Vous pouvez utiliser cette clé de condition pour vous assurer que la ressource à laquelle vous accédez appartient au compte spécifié dans AWS Organizations.

Dans certains cas, vous devrez peut-être autoriser l'accès à des ressources AWS détenues, à des ressources qui n'appartiennent pas à votre organisation et auxquelles ont accès vos mandants ou des AWS services agissant en votre nom. Pour plus d'informations sur ces scénarios, voir Établissement d'un périmètre de données sur AWS : Autoriser uniquement les ressources fiables de mon organisation.

Périmètre du réseau

Un périmètre réseau est un ensemble de contrôles d'accès préventifs grossiers qui permettent de garantir que vos identités ne peuvent accéder aux ressources qu'à partir des réseaux attendus et que vos ressources ne sont accessibles qu'à partir des réseaux attendus. Les réseaux attendus incluent vos centres de données sur site, vos clouds privés virtuels (VPC) et les réseaux de AWS services agissant en votre nom.

Les clés de condition globales suivantes permettent de renforcer les contrôles du périmètre du réseau. Utilisez ces clés dans les politiques de contrôle des services (SCP) pour restreindre les réseaux à partir desquels vos identités peuvent communiquer, ou dans les politiques basées sur les ressources pour restreindre l'accès aux ressources aux réseaux attendus.

- lois : SourceIp— Vous pouvez utiliser cette clé de condition pour vous assurer que l'adresse IP du demandeur se situe dans une plage d'adresses IP spécifiée.
- lois : SourceVpc— Vous pouvez utiliser cette clé de condition pour vous assurer que le point de terminaison du VPC par lequel transite la demande appartient au VPC spécifié.
- lois : SourceVpce— Vous pouvez utiliser cette clé de condition pour vous assurer que la demande passe par le point de terminaison VPC spécifié.
- AWS : via AWSService— Vous pouvez utiliser cette clé de condition pour vous assurer que vous Services AWS pouvez faire des demandes au nom de votre principal utilisateur Transmission des sessions d'accès (FAS).

- lois : PrincipalIs AWSService— Vous pouvez utiliser cette clé de condition pour vous assurer que vous Services AWS pouvez accéder à vos ressources en utilisantAWS principes de service.

Il existe d'autres scénarios dans lesquels vous devez autoriser l'accès Services AWS à ces ressources depuis l'extérieur de votre réseau. Pour plus d'informations, voir Établissement d'un périmètre de données sur AWS : Autoriser l'accès aux données de l'entreprise uniquement à partir des réseaux attendus.

Ressources pour en savoir plus sur les périmètres de données

Les ressources suivantes peuvent vous aider à en savoir plus sur les périmètres de données AWS transversaux.

- Périmètres de données activés AWS : découvrez les périmètres de données, leurs avantages et leurs cas d'utilisation.
- Livre blanc : Building a Data Perimeter on AWS — Ce document décrit les meilleures pratiques et les services disponibles pour créer un périmètre autour de vos identités, de vos ressources et de vos réseaux dans. AWS
- Webinaire : Création d'un périmètre de données dans AWS — Découvrez où et comment mettre en œuvre des contrôles du périmètre des données en fonction de différents scénarios de risque.
- Série d'articles de blog : Établissement d'un périmètre de données sur AWS — Ces articles de blog fournissent des conseils prescriptifs sur l'établissement de votre périmètre de données à grande échelle, y compris les principales considérations relatives à la sécurité et à la mise en œuvre.
- Exemples de politiques de périmètre de données — Ce GitHub référentiel contient des exemples de politiques qui couvrent certains modèles courants pour vous aider à implémenter un périmètre de données sur AWS.
- Data Perimeter Helper : cet outil vous aide à concevoir et à anticiper l'impact de vos contrôles de périmètre de données en analysant les activités d'accès dans vos AWS CloudTrailjournaux.

Limites d'autorisations pour les entités IAM

AWS prend en charge les limites d'autorisations pour les entités IAM (utilisateurs ou rôles). Une limite d'autorisations est une fonctionnalité avancée permettant d'utiliser une politique gérée pour définir les autorisations maximales qu'une politique basée sur l'identité peut accorder à une entité IAM. La limite d'autorisations d'une entité lui permet d'effectuer uniquement les actions autorisées à la fois par ses politiques basées sur l'identité et ses limites d'autorisations.

Pour de plus amples informations sur les types de politiques, veuillez consulter Types de politique.

> **Important**
>
> N'utilisez pas de déclarations de politique basée sur les ressources qui incluent un élément de politique `NotPrincipal` ayant un effet `Deny` sur les utilisateurs IAM ou les rôles auxquels est attachée une politique de limite des autorisations. L'élément `NotPrincipal` ayant un effet `Deny` refusera toujours tout principal IAM auquel est attachée une politique de limite des autorisations, quelles que soient les valeurs spécifiées dans l'élément `NotPrincipal`. Certains utilisateurs ou rôles IAM qui auraient autrement eu accès à la ressource n'y ont donc plus accès. Nous vous recommandons de modifier vos déclarations de politique basée sur les ressources afin d'utiliser l'opérateur de condition `ArnNotEquals` avec la clé de contexte `aws:PrincipalArn` dans le but de limiter l'accès plutôt que l'élément `NotPrincipal`. Pour en savoir plus sur l'élément `NotPrincipal`, veuillez consulter AWS Éléments de politique JSON : NotPrincipal.

Vous pouvez utiliser une politique AWS gérée ou une politique gérée par le client pour définir les limites d'une entité IAM (utilisateur ou rôle). Cette politique limite le nombre maximum d'autorisations pour l'utilisateur ou le rôle.

Supposons, par exemple, que l'utilisateur IAM nommé `ShirleyRodriguez` soit autorisé à gérer uniquement Amazon S3 CloudWatch, Amazon et Amazon EC2. Pour appliquer cette règle, vous pouvez utiliser la politique suivante afin de définir les autorisations pour l'utilisateur `ShirleyRodriguez` :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

Lorsque vous utilisez une politique pour définir la limite d'autorisations d'un utilisateur, elle limite les autorisations de l'utilisateur mais ne fournit pas seule les autorisations. Dans cet exemple, la politique définit les autorisations maximales pour toutes `ShirleyRodriguez` les opérations dans Amazon S3 et Amazon EC2. CloudWatch Shirley ne peut pas exécuter d'opérations dans les autres services, y compris IAM, même si elle dispose d'une politique d'autorisations qui le permet. Par exemple, vous pouvez ajouter la politique suivante à l'utilisateur `ShirleyRodriguez` :

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

Cette politique permet de créer un utilisateur dans IAM. Si vous attachez cette politique d'autorisation à l'utilisateur `ShirleyRodriguez` et que Shirley tente de créer un utilisateur, l'opération échoue. Elle échoue car la limite des autorisations n'autorise pas l'opération `iam:CreateUser`. Compte tenu de ces deux politiques, Shirley n'est pas autorisée à effectuer des opérations dans AWS. Vous devez ajouter une politique d'autorisations différente pour autoriser des actions dans d'autres services, tels qu'Amazon S3. Sinon, vous pouvez mettre à jour la limite d'autorisations pour lui permettre de créer un utilisateur dans IAM.

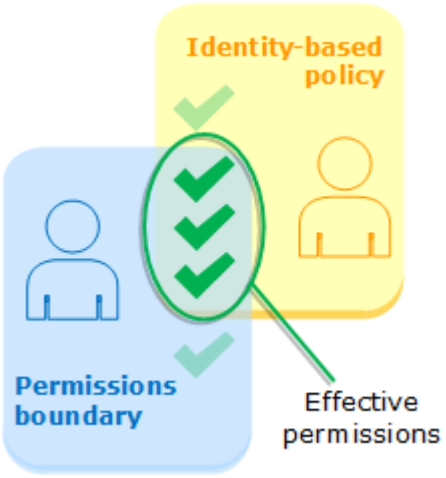
Évaluation des autorisations effectives avec limites

La limite d'autorisations pour une entité IAM (utilisateur ou rôle) définit le nombre maximum d'autorisations maximum dont peut disposer une entité. Cela peut modifier les autorisations effectives pour cet utilisateur ou rôle. Les autorisations effectives d'une entité correspondent aux autorisations accordées par toutes les politiques affectant l'utilisateur ou le rôle. Dans un compte, les autorisations d'une entité peuvent être affectées par des politiques basées sur l'identité, des politiques basées sur les ressources, des limites d'autorisations, des politiques de contrôle de service Organizations ou des politiques de session. Pour de plus amples informations sur les différents types de politiques, veuillez consulter Politiques et autorisations dans IAM.

Si l'un de ces types de politique refuse explicitement l'accès pour une opération, la demande est refusée. Les autorisations accordées à une entité par plusieurs types d'autorisations sont plus

complexes. Pour plus de détails sur le mode AWS d'évaluation des politiques, consultezLogique d'évaluation de politiques.

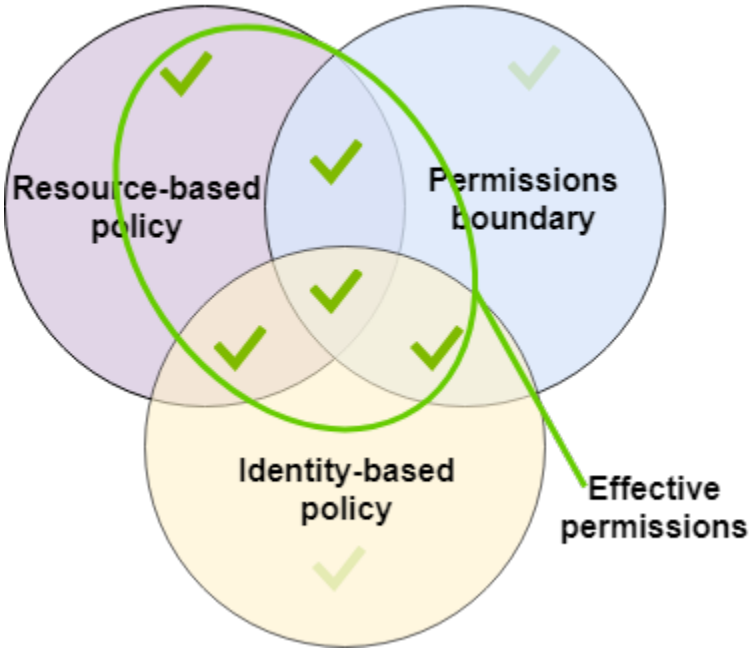
Identity-based policies with boundaries (Politiques basées sur l'identité avec des limites) : les politiques basées sur l'identité sont des politiques en ligne ou gérées qui sont attachées à un utilisateur, un groupe d'utilisateurs ou un rôle. Les politiques basées sur l'identité accordent une autorisation à l'entité, et les limites d'autorisations limitent ces autorisations. Les autorisations effectives sont une combinaison de ces deux types de politique. Un refus explicite dans l'une ou l'autre de ces stratégies remplace l'autorisation.



Resource-based policies (Politiques basées sur les ressources) : les politiques basées sur les ressources contrôlent la façon dont le principal spécifié peut accéder à la ressource à laquelle la politique est attachée.

Politiques basées sur des ressources pour les utilisateurs IAM

> Au sein d'un même compte, les politiques basées sur les ressources qui accordent des autorisations à un ARN d'utilisateur IAM (qui n'est pas une séance d'utilisateur fédéré) ne sont pas limitées par un rejet implicite dans une politique basée sur l'identité ou une limite d'autorisations.



Politiques basées sur les ressources pour les rôles IAM

Rôle IAM – Les politiques basées sur les ressources qui accordent des autorisations à un ARN de rôle IAM sont limitées par un rejet implicite dans une limite d'autorisations ou une politique de séance.

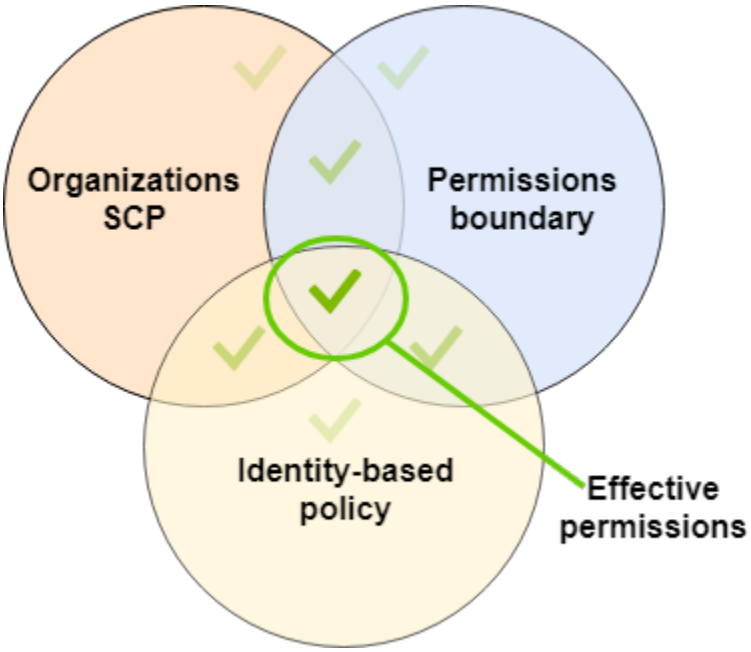
Séance de rôle IAM – Au sein d'un même compte, les politiques basées sur les ressources qui accordent des autorisations à un ARN de séance de rôle IAM accordent des autorisations directement à la séance de rôle endossée. Les autorisations accordées directement à une séance ne sont pas limitées par un rejet implicite dans une politique basée sur l'identité, une limite d'autorisations ou une politique de séance. Lorsque vous endossez un rôle et faites une demande, le principal qui fait la demande est l'ARN de séance du rôle IAM et non l'ARN du rôle lui-même.

Politiques basées sur les ressources pour les sessions d'utilisateur fédéré IAM

Séances d'utilisateur fédéré IAM – Une séance d'utilisateur fédéré IAM est une séance créée en appelant les outils `GetFederationToken`. Lorsqu'un utilisateur fédéré fait une demande, le principal qui effectue la demande est l'ARN d'utilisateur fédéré et non l'ARN de l'utilisateur IAM qui a fédéré. Dans le même compte, les politiques basées sur les ressources accordant des autorisations à un ARN d'utilisateur fédéré accordent des autorisations directement à la séance. Les autorisations accordées directement à une séance ne sont pas limitées par un rejet implicite dans une politique basée sur l'identité, une limite d'autorisations ou une politique de séance.

Cependant, si une politique basée sur les ressources accorde une autorisation à l'ARN de l'utilisateur IAM qui s'est fédéré, les demandes faites par l'utilisateur fédéré pendant la séance sont limitées par un rejet implicite dans une limite d'autorisation ou une politique de séance.

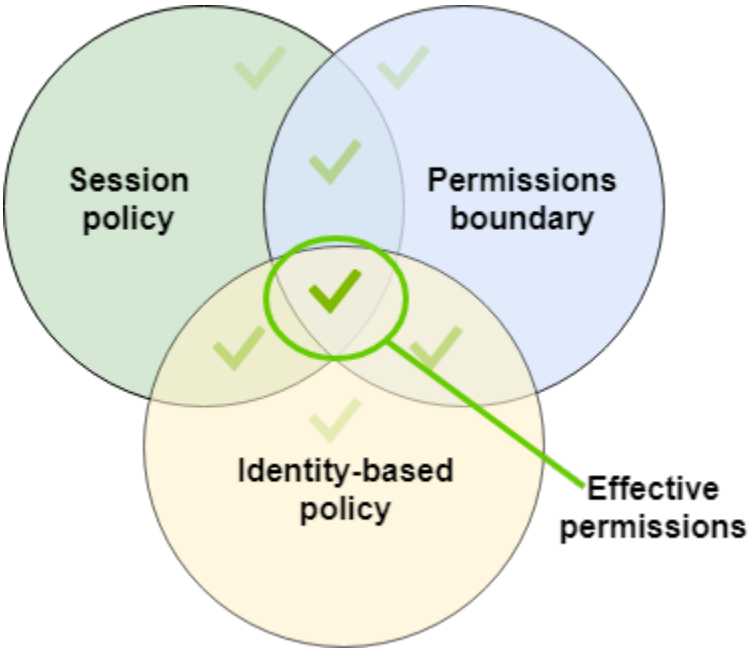
SCP des organisations : les politiques de contrôle de service (Service Control Policies, SCP) sont appliquées à l'ensemble d'un Compte AWS. Elles limitent les autorisations pour chaque demande faite par un principal dans le compte. Une entité IAM (utilisateur ou rôle) peut effectuer une demande qui est soumise à une politique de contrôle de service, une limite d'autorisations et une politique basée sur l'identité. Dans ce cas, la demande est autorisée uniquement si les trois types de politique l'autorisent. Les autorisations effectives sont la combinaison des trois types de politique. Un refus explicite dans l'une de ces politiques annule l'autorisation.



Vous pouvez savoir si votre compte est membre d'une organisation dans AWS Organizations. Les membres de l'organisation pourraient être affectés par une SCP. Pour afficher ces données à l'aide de la AWS CLI commande ou de AWS l'opération d'API, vous devez disposer des autorisations nécessaires à l'`organizations:DescribeOrganization`action pour votre entité Organizations. Vous devez disposer des autorisations supplémentaires pour effectuer l'opération dans la console Organizations. Pour savoir si un SCP refuse l'accès à une demande spécifique ou pour modifier vos autorisations effectives, contactez votre AWS Organizations administrateur.

Politiques de session : les politiques de session sont des politiques avancées que vous passez en tant que paramètre lorsque vous programmez afin de créer une session temporaire pour un rôle ou un utilisateur fédéré. Les autorisations pour une session proviennent de l'entité IAM (utilisateur

ou rôle) utilisée pour créer la session et de la politique de la session. Les autorisations de politique basée sur l'identité de l'entité sont limitées par la politique de la session et la limite d'autorisations. Les autorisations effectives pour cet ensemble de types de politique sont la combinaison des trois types de politique. Un refus explicite dans l'une de ces politiques annule l'autorisation. Pour de plus amples informations sur les politiques de session, veuillez consulter Politiques de session.



Délégation de responsabilité à d'autres utilisateurs à l'aide des limites d'autorisations

Vous pouvez utiliser des limites d'autorisations pour déléguer à des utilisateurs IAM de votre compte des tâches de gestion d'autorisations, comme la création d'utilisateurs,. Cela permet à d'autres utilisateurs d'exécuter des tâches en votre nom dans une limite d'autorisations spécifique.

Supposons, par exemple, que María soit l'administratrice de la X-Company Compte AWS. Elle souhaite déléguer des tâches de création d'utilisateurs à Zhang. Toutefois, elle doit s'assurer que Zhang crée des utilisateurs qui respectent les règles de l'entreprise suivantes :

- Les utilisateurs ne peuvent pas utiliser IAM pour créer ou gérer des utilisateurs, groupes, rôles ou politiques.
- Les utilisateurs se voient refuser l'accès au compartiment `logs` Amazon S3 et ne peuvent pas accéder à l'instance Amazon EC2 `i-1234567890abcdef0`.
- Les utilisateurs ne peuvent pas supprimer leurs propres politiques de limite.

Pour appliquer ces règles, María exécute les tâches suivantes dont les détails sont inclus ci-dessous :

1. María crée la politique gérée `XCompanyBoundaries` pour utiliser en tant que limite d'autorisations pour tous les nouveaux utilisateurs du compte.
2. María crée la politique gérée `DelegatedUserBoundary` et l'assigne en tant que limite d'autorisations pour Zhang. Maria prend note de l'ARN de son administrateur et l'utilise dans la stratégie pour empêcher Zhang d'y accéder.
3. María crée la politique gérée `DelegatedUserPermissions` et l'attache en tant que limite d'autorisations pour Zhang.
4. María explique à Zhang ses nouvelles responsabilités et limites.

Tâche 1 : María doit d'abord créer une politique gérée pour définir la limite pour les nouveaux utilisateurs. María autorise Zhang à donner aux utilisateurs les politiques d'autorisations dont ils ont besoin, mais elle souhaite que ces derniers soient limités. Pour ce faire, elle crée la politique gérée par le client suivante avec le nom `XCompanyBoundaries`. Cette politique effectue les opérations suivantes :

- Permet aux utilisateurs d'avoir un accès complet à plusieurs services
- Autorise un accès autonome limité dans la console IAM. Cela signifie qu'ils peuvent changer leur mot de passe après s'être connecté à la console. Ils ne peuvent pas définir leur mot de passe initial. Pour que cela soit possible, ajoutez l'action `"*LoginProfile"` à l'instruction `AllowManageOwnPasswordAndAccessKeys`.
- Refuse aux utilisateurs l'accès au compartiment des journaux Amazon S3 ou à l'instance Amazon EC2 `i-1234567890abcdef0`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ServiceBoundaries",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*",
                "dynamodb:*"
```

```
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
}
```

Chaque instruction répond à un objectif distinct :

1. L'`ServiceBoundaries`énoncé de cette politique permet un accès complet aux AWS services spécifiés. Cela signifie que de nouvelles actions de l'utilisateur dans ces services sont uniquement limitées par les politiques d'autorisations qui lui sont attachées.
2. L'instruction `AllowIAMConsoleForCredentials` autorise l'accès pour répertorier tous les utilisateurs IAM. Cet accès est nécessaire pour accéder à la page Utilisateurs dans la AWS Management Console. Il permet également d'afficher les exigences de mot de passe pour le compte, qui sont nécessaires lorsque vous modifiez votre mot de passe.
3. La déclaration `AllowManageOwnPasswordAndAccessKeys` autorise les utilisateurs à gérer uniquement leur propre mot de passe de console et les clés d'accès par programmation. Ceci est important si Zhang ou un autre administrateur attribue à un nouvel utilisateur une politique d'autorisations avec un accès complet à IAM. Dans ce cas, cet utilisateur peut ensuite modifier ses propres autorisations ou celles des autres utilisateurs. Cette instruction évite ce genre de problème.
4. L'instruction `DenyS3Logs` refuse explicitement l'accès au compartiment `logs`.
5. L'instruction `DenyEC2Production` refuse explicitement l'accès à l'instance `i-1234567890abcdef0`.

Tâche 2 : María souhaite autoriser Zhang à créer tous les utilisateurs X-Company, mais uniquement avec la limite d'autorisations `XCompanyBoundaries`. Elle crée la politique gérée par le client suivante et la nomme `DelegatedUserBoundary`. Cette politique définit le nombre maximum de permissions dont dispose Zhang.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOrChangeOnlyWithBoundary",
            "Effect": "Allow",
            "Action": [
                "iam:AttachUserPolicy",
                "iam:CreateUser",
                "iam:DeleteUserPolicy",
                "iam:DetachUserPolicy",
                "iam:PutUserPermissionsBoundary",
                "iam:PutUserPolicy"
            ],
            "Resource": "*",
            "Condition": {
```

```
               "StringEquals": {
                 "iam:PermissionsBoundary": "arn:aws:iam::123456789012:policy/
XCompanyBoundaries"
                }
            }
        },
        {
            "Sid": "CloudWatchAndOtherIAMTasks",
            "Effect": "Allow",
            "Action": [
              "cloudwatch:*",
              "iam:CreateAccessKey",
              "iam:CreateGroup",
              "iam:CreateLoginProfile",
              "iam:CreatePolicy",
              "iam:DeleteGroup",
              "iam:DeletePolicy",
              "iam:DeletePolicyVersion",
              "iam:DeleteUser",
              "iam:GetAccountPasswordPolicy",
              "iam:GetGroup",
              "iam:GetLoginProfile",
              "iam:GetPolicy",
              "iam:GetPolicyVersion",
              "iam:GetRolePolicy",
              "iam:GetUser",
              "iam:GetUserPolicy",
              "iam:ListAccessKeys",
              "iam:ListAttachedRolePolicies",
              "iam:ListAttachedUserPolicies",
              "iam:ListEntitiesForPolicy",
              "iam:ListGroups",
              "iam:ListGroupsForUser",
              "iam:ListMFADevices",
              "iam:ListPolicies",
              "iam:ListPolicyVersions",
              "iam:ListRolePolicies",
              "iam:ListSSHPublicKeys",
              "iam:ListServiceSpecificCredentials",
              "iam:ListSigningCertificates",
              "iam:ListUserPolicies",
              "iam:ListUsers",
              "iam:SetDefaultPolicyVersion",
              "iam:SimulateCustomPolicy",
```

```
                "iam:SimulatePrincipalPolicy",
                "iam:UpdateGroup",
                "iam:UpdateLoginProfile",
                "iam:UpdateUser"
            ],
            "NotResource": "arn:aws:iam::123456789012:user/Maria"
        },
        {
            "Sid": "NoBoundaryPolicyEdit",
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
                "iam:DeletePolicy",
                "iam:DeletePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": [
                "arn:aws:iam::123456789012:policy/XCompanyBoundaries",
                "arn:aws:iam::123456789012:policy/DelegatedUserBoundary"
            ]
        },
        {
            "Sid": "NoBoundaryUserDelete",
            "Effect": "Deny",
            "Action": "iam:DeleteUserPermissionsBoundary",
            "Resource": "*"
        }
    ]
}
```

Chaque instruction répond à un objectif distinct :

1. L'instruction `CreateOrChangeOnlyWithBoundary` autorise Zhang à créer des utilisateurs IAM, mais uniquement s'il utilise la politique `XCompanyBoundaries` permettant de définir la limite d'autorisations. Cette instruction lui permet également de définir la limite d'autorisations pour les utilisateurs existants, mais uniquement à l'aide de cette même politique. Enfin, cette instruction autorise Zhang à gérer des politiques d'autorisations pour les utilisateurs avec cette limite d'autorisations définie.
2. L'instruction `CloudWatchAndOtherIAMTasks` autorise Zhang à exécuter les tâches de gestion d'autres utilisateurs, groupes et politiques. Il possède les autorisations de réinitialiser les mots de passe et de créer des clés d'accès pour tout utilisateur IAM non répertorié dans l'élément de

politique `NotResource`. Cela lui permet d'aider les utilisateurs qui rencontrent des problèmes de connexion.

3. L'instruction `NoBoundaryPolicyEdit` refuse l'accès à Zhang pour mettre à jour la politique `XCompanyBoundaries`. Il n'est pas autorisé à modifier une politique utilisée pour définir la limite d'autorisations pour lui-même ou d'autres utilisateurs.
4. L'instruction `NoBoundaryUserDelete` interdit à Zhang de supprimer la limite d'autorisations pour lui-même ou d'autres utilisateurs.

María attribue ensuite la politique `DelegatedUserBoundary` comme limite d'autorisations pour l'utilisateur `Zhang`.

Tâche 3 : María doit créer une politique d'autorisations pour Zhang, car la limite d'autorisations restreint le nombre maximum d'autorisations, mais n'accorde pas à elle seule l'accès. Elle crée la politique suivante et la nomme `DelegatedUserPermissions`. Cette politique définit les opérations que Zhang peut exécuter au sein de la limite définie.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "IAM",
            "Effect": "Allow",
            "Action": "iam:*",
            "Resource": "*"
        },
        {
            "Sid": "CloudWatchLimited",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:GetDashboard",
                "cloudwatch:GetMetricData",
                "cloudwatch:ListDashboards",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3BucketContents",
            "Effect": "Allow",
```

```
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::ZhangBucket"
        }
    ]
}
```

Chaque instruction répond à un objectif distinct :

1. L'instruction `IAM` de la politique accorde à Zhang un accès complet à IAM. Toutefois, ses autorisations IAM effectives sont uniquement restreintes par sa limite d'autorisations, car cette dernière autorise uniquement certaines opérations IAM.
2. La `CloudWatchLimited` déclaration permet à Zhang d'effectuer cinq actions dans CloudWatch. Sa limite d'autorisations autorise toutes les actions CloudWatch, de sorte que ses CloudWatch autorisations effectives ne sont limitées que par sa politique d'autorisations.
3. L'instruction `S3BucketContents` autorise Zhang à répertorier le compartiment Amazon S3 `ZhangBucket`. Toutefois, sa limite d'autorisations n'autorise aucune action Amazon S3. Par conséquent, il ne peut pas exécuter d'opérations S3, quelle que soit sa politique d'autorisations.

> **Note**
>
> Les politiques de Zhang lui permettent de créer un utilisateur qui peut alors accéder à des ressources Amazon S3 auxquelles il ne peut pas accéder. En déléguant ces actions administratives, Maria approuve l'accès de Zhang à Amazon S3.

María assigne ensuite la politique `DelegatedUserPermissions` en tant que politique d'autorisations pour l'utilisateur `Zhang`.

Tâche 4 : Elle explique à Zhang comment créer un nouvel utilisateur. Elle lui indique qu'il peut créer de nouveaux utilisateurs avec les autorisations dont il a besoin, mais il doit leur assigner la politique `XCompanyBoundaries` en tant que limite d'autorisations.

Zhang exécute les tâches suivantes :

1. Zhang crée un utilisateur avec l' AWS Management Console. Il saisit le nom d'utilisateur `Nikhil` et accorde à l'utilisateur l'accès à la console. Il décoche la case en regard de Requires password reset (Réinitialisation du mot de passe requise), car les politiques ci-dessus permettent à Zhang de modifier son mot de passe uniquement après s'être connecté à la console IAM.

2. Sur la page Définir les autorisations, Zhang choisit les politiques d'autorisation IAM FullAccess et AmazonS3 ReadOnlyAccess qui permettent à Nikhil de faire son travail.
3. Zhang ignore la section Set permissions boundary (Définir une limite d'autorisations), oubliant ainsi les instructions de María.
4. Zhang examine les détails de l'utilisateur et choisit Créer un utilisateur.

   L'opération échoue et l'accès est refusé. La limite d'autorisations `DelegatedUserBoundary` de Zhang exige que tous les utilisateurs qu'il crée utilisent la politique `XCompanyBoundaries` en tant que limite d'autorisations.
5. Zhang revient à la page précédente. Dans la section Set permissions boundary (Définir une limite d'autorisations), il choisit la politique `XCompanyBoundaries`.
6. Zhang examine les détails de l'utilisateur et choisit Créer un utilisateur.

   L'utilisateur est créé.

Lorsque Nikhil se connecte, il accède à IAM et Amazon S3, à l'exception des opérations refusées par la limite d'autorisations. Par exemple, il peut modifier son propre mot de passe dans IAM, mais ne peut pas créer d'autre utilisateur ou modifier ses politiques. Nikhil a un accès en lecture seule à Amazon S3.

Si une personne ajoute au compartiment `logs` une politique basée sur les ressources qui autorise Nikhil à placer un objet dans le compartiment, il ne peut toujours pas accéder au compartiment. En effet, toutes les actions sur le compartiment `logs` sont explicitement refusées par sa limite d'autorisations. Un refus explicite dans n'importe quel type de politique se traduit par le refus d'une demande. Toutefois, si une politique basée sur les ressources attachée à un secret Secrets Manager permet à Nikhil d'effectuer l'action `secretsmanager:GetSecretValue`, Nikhil peut récupérer et déchiffrer le secret. En effet, les opérations Secrets Manager ne sont pas explicitement refusées par sa limite d'autorisations et les conditions de refus implicite dans les limites d'autorisations ne limitent pas les politiques basées sur les ressources.

Politiques basées sur l'identité et Politiques basées sur une ressource

Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit leurs autorisations. Lorsque vous créez une politique d'autorisations pour limiter l'accès à une ressource, vous pouvez choisir une politique basée sur l'identité ou une politique basée sur les ressources.

Les politiques basées sur une identité sont attachées à un utilisateur, un groupe ou un rôle IAM. Ces politiques vous permettent de spécifier ce que peut faire cette identité (ses autorisations). Par exemple, vous pouvez attacher la politique à l'utilisateur IAM nommé John, en indiquant qu'il est autorisé à effectuer l'action Amazon EC2 `RunInstances`. La politique peut aussi indiquer que John est autorisé à extraire des éléments d'une table Amazon DynamoDB nommée `MyCompany`. Vous pouvez également autoriser John à gérer ses propres informations d'identification de sécurité IAM. Les politiques basées sur une identité peuvent être gérées ou en ligne.

Les politiques basées sur les ressources sont attachées à une ressource. Par exemple, vous pouvez associer des politiques basées sur les ressources aux compartiments Amazon S3, aux files d'attente Amazon SQS, aux points de terminaison VPC, aux clés de AWS Key Management Service chiffrement, ainsi qu'aux tables et aux flux Amazon DynamoDB. Pour obtenir la liste des services qui prennent en charge les politiques basées sur une ressource, consultez AWS services qui fonctionnent avec IAM.

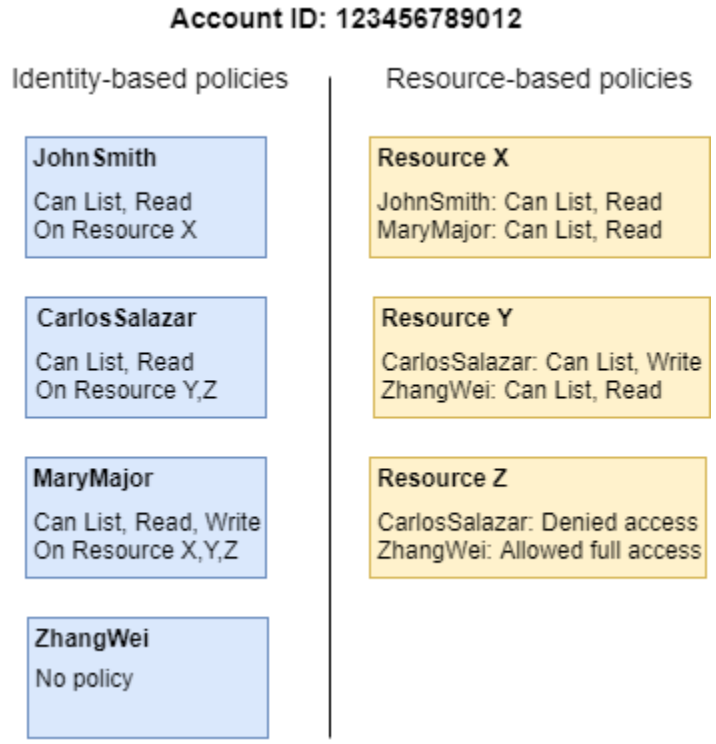
Avec les politiques basées sur une ressource, vous pouvez spécifier qui a accès à la ressource et quelles actions peuvent être exécutées. Pour savoir si les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, consultez Qu'est-ce qu'IAM Access Analyzer ?. Les autorisations basées sur une ressource sont en ligne uniquement, pas gérées.

> **Note**
>
> Les politiques basées sur les ressources diffèrent des autorisations de niveau ressource. Vous pouvez attacher directement à une ressource les politiques basées sur les ressources, comme décrit dans cette rubrique. Les autorisations de niveau ressource font référence à la possibilité d'utiliser les ARN pour spécifier des ressources spécifiques dans une politique. Les politiques basées sur les ressources ne sont prises en charge que par certains AWS services. Pour obtenir la liste des services qui prennent en charge les politiques basées sur les ressources et les autorisations de niveau ressource, consultez AWS services qui fonctionnent avec IAM.

Pour découvrir comment les politiques basées sur l'identité et les politiques basées sur les ressources interagissent dans un même compte, veuillez consulter Évaluation des politiques dans un compte unique.

Pour découvrir comment les politiques interagissent entre comptes, veuillez consulter Logique d'évaluation des politiques entre comptes.

Pour mieux comprendre ces concepts, voir la figure suivante. L'administrateur des `123456789012` politiques basées sur l'identité attachée au compte pour les utilisateurs `JohnSmith`, `CarlosSalazar` et `MaryMajor`. Certaines actions de ces politiques peuvent être effectuées sur des ressources spécifiques. Par exemple, l'utilisateur `JohnSmith` peut effectuer certaines actions sur `Resource X`. Il s'agit d'une autorisation au niveau ressource dans une politique basée sur une identité. L'administrateur a également ajouté les politiques basées sur les ressources pour `Resource X`, `Resource Y` et `Resource Z`. Les politiques basées sur les ressources vous permettent de spécifier qui peut accéder à cette ressource. Par exemple, la politique basée sur les ressources sur `Resource X` accorde aux utilisateurs `JohnSmith` et `MaryMajor` l'accès en lecture à la ressource.



L'exemple de compte `123456789012` autorise les utilisateurs suivants à exécuter les actions répertoriées :

- JohnSmith— John peut exécuter des actions de liste et de lecture sur`Resource X`. Cette autorisation lui est accordée par la politique basée sur l'identité sur son utilisateur et par la politique basée sur les ressources sur `Resource X`.
- CarlosSalazar— Carlos peut exécuter des actions de liste, de lecture et d'écriture`Resource Y`, mais il n'y a pas accès`Resource Z`. La politique basée sur l'identité sur Carlos lui permet

d'effectuer les actions d'affichage et de lecture sur `Resource Y`. La politique basée sur la ressource `Resource Y` lui accorde également les autorisations d'écriture. Cependant, même si sa politique basée sur les identités lui permet l'accès à `Resource Z`, la politique basée sur les ressources `Resource Z` lui refuse l'accès. Un `Deny` explicite remplace une `Allow` et l'accès à `Resource Z` est refusé. Pour plus d'informations, consultez Logique d'évaluation de politiques.

- MaryMajor— Mary peut effectuer des opérations de liste, de lecture et d'écriture sur `Resource X`Resource Y, et`Resource Z`. Sa politique basée sur les identités lui permet plus d'actions sur plus de ressources que les politiques basées sur les ressources, mais aucune d'elles ne lui refuse l'accès.
- ZhangWei— Zhang a un accès complet à`Resource Z`. Zhang n'a pas de politiques basées sur l'identité, mais la politique basée sur les `Resource Z` ressources l'autorise un accès complet à la ressource. Zhang peut également effectuer les actions List (Liste) et Read (Lire) sur `Resource Y`.

Les politiques basées sur l'identité et les politiques basées sur les ressource sont les deux politiques d'autorisations et sont évaluées ensemble. Pour une demande à laquelle seules les politiques d'autorisation s'appliquent, AWS vérifiez d'abord toutes les politiques pour un`Deny`. Le cas échéant, la demande est refusée. Ensuite, AWS vérifie chaque `Allow`. Si au moins une instruction de politique autorise l'action dans la demande, la demande est autorisée. Il n'est pas important que l'`Allow` soit dans la politique basée sur les identités ou dans celle basée sur les ressources.

> **Important**
>
> Cette logique s'applique uniquement lorsque la demande est effectuée au sein d'un même Compte AWS. Pour les demandes effectuées à partir d'un compte vers un autre, le demandeur `Account A` doit avoir une politique basée sur une identité qui leur permet d'adresser une demande à la ressource dans `Account B`. En outre, la politique basée sur les ressources dans `Account B` doivent autoriser le demandeur dans `Account A` à accéder à la ressource. Il doit y avoir des politiques dans les deux comptes qui autorisent l'opération, sinon la demande échoue. Pour plus d'informations sur l'utilisation des politiques basées sur une ressource pour l'accès inter-compte, consultez Accès intercompte aux ressources dans IAM.

Un utilisateur qui dispose d'autorisations spécifiques peut demander une ressource à laquelle une politique d'autorisation est attachée. Dans ce cas, AWS évalue les deux ensembles d'autorisations

pour déterminer s'il convient d'accorder l'accès à la ressource. Pour plus d'informations sur la manière dont les politiques sont évaluées, consultez Logique d'évaluation de politiques.

> **Note**
>
> Amazon S3 prend en charge les politiques basées sur l'identité et les politiques basées sur les ressources (appelées politiques de compartiment). En outre, Amazon S3 prend en charge un mécanisme d'autorisation appelé liste de contrôle d'accès (ACL) qui est indépendant des politiques et des autorisations IAM. Vous pouvez utiliser des politiques IAM combinées à des listes ACL (liste de contrôle d'accès) Amazon S3. Pour plus d'informations, veuillez consulter contrôle d'accès dans le guide de l'utilisateur du service de stockage simple Amazon.

Contrôle de l'accès aux AWS ressources à l'aide de politiques

Vous pouvez utiliser une politique pour contrôler l'accès aux ressources au sein d'IAM ou de la totalité d' AWS entre elles.

Pour utiliser une politique visant à contrôler l'accès AWS, vous devez comprendre comment AWS octroyer l'accès. AWS est composé de collections de ressources. Un utilisateur IAM est une ressource. Un compartiment Amazon S3 est une ressource. Lorsque vous utilisez l' AWS API, le AWS CLI, ou le AWS Management Console pour effectuer une opération (telle que la création d'un utilisateur), vous envoyez une demande pour cette opération. Votre demande spécifie une action, une ressource, une entité du principal (utilisateur ou rôle), un compte du principal et toutes les informations nécessaires relatives à la demande. Toutes ces informations fournissent le contexte.

AWS vérifie ensuite que vous (le principal) êtes authentifié (connecté) et autorisé (autorisé) à effectuer l'action spécifiée sur la ressource spécifiée. Lors de l'autorisation, AWS vérifie toutes les politiques qui s'appliquent au contexte de votre demande. La plupart des politiques sont stockées AWS sous forme de documents JSON et spécifient les autorisations pour les entités principales. Pour plus d'informations sur les types de politiques et les utilisations, veuillez consulter Politiques et autorisations dans IAM.

AWS autorise la demande uniquement si chaque partie de votre demande est autorisée par les politiques. Pour afficher un schéma de ce processus, reportez-vous à Fonctionnement de IAM. Pour plus de détails sur AWS la manière de déterminer si une demande est autorisée, consultezLogique d'évaluation de politiques.

Lorsque vous créez une politique IAM, vous pouvez contrôler l'accès aux éléments suivants :

- Entités du principal – Contrôler ce que l'individu à l'origine de la requête (le principal) est autorisé à faire.
- IAM Identities (Identités IAM) : contrôlez à quelles identités IAM (groupes d'utilisateurs, utilisateurs et rôles) il est possible d'accéder et comment y accéder.
- IAM Policies (Politiques IAM) : contrôlez qui peut créer, modifier et supprimer les politiques gérées par les clients, et qui peut attacher et détacher toutes les politiques gérées.
- AWS Resources (Ressources AWS) : contrôlez qui a accès aux ressources à l'aide d'une politique basée sur les identités ou sur les ressources.
- AWS Accounts (Comptes) : contrôle si une requête est autorisée uniquement pour les membres d'un compte spécifique.

Les politiques vous permettent de spécifier qui a accès aux AWS ressources et quelles actions ils peuvent effectuer sur ces ressources. Chaque utilisateur IAM démarre sans autorisation. En d'autres termes, par défaut, les utilisateurs ne peuvent rien faire, pas même afficher leurs propres clés d'accès. Pour autoriser un utilisateur à effectuer une action, vous pouvez ajouter l'autorisation appropriée pour l'utilisateur (c'est-à-dire attacher une politique à celui-ci). Vous pouvez également ajouter l'utilisateur à un groupe d'utilisateurs disposant de l'autorisation prévue.

Par exemple, vous pourriez accorder à un utilisateur l'autorisation d'afficher ses propres clés d'accès. Vous pourriez également étendre cette autorisation et permettre également à chaque utilisateur de créer, mettre à jour et la supprimer ses propres clés.

Lorsque vous accordez des autorisations à un groupe d'utilisateurs, tous les utilisateurs de ce groupe obtiennent ces autorisations. Par exemple, vous pouvez autoriser le groupe d'utilisateurs Administrateurs à effectuer n'importe quelle action IAM sur n'importe quelle Compte AWS ressource. Un autre exemple : vous pouvez donner au groupe d'utilisateurs Managers (Gestionnaires) l'autorisation de décrire les instances Amazon EC2 de l' Compte AWS.

Pour de plus amples informations sur la façon de déléguer des autorisations de base à vos utilisateurs, groupes d'utilisateurs et rôles, veuillez consulter Autorisations requises pour accéder aux autres ressources IAM. Pour obtenir des exemples supplémentaires de politiques illustrant des autorisations de base, consultez Exemples de politiques de gestion de ressources IAM.

Contrôle de l'accès pour les entités du principal

Vous pouvez utiliser des politiques pour contrôler ce que la personne à l'origine de la demande (le principal) est autorisée à effectuer. Pour ce faire, vous devez attacher à l'identité de cette personne

une politique basée sur l'identité (utilisateur, groupe d'utilisateurs ou rôle). Vous pouvez également utiliser une limite d'autorisations pour définir les autorisations maximales qu'une entité (utilisateur ou rôle) peut avoir.

Supposons, par exemple, que vous souhaitiez que l'utilisateur Zhang Wei ait un accès complet à Amazon DynamoDB CloudWatch, Amazon EC2 et Amazon S3. Vous pouvez créer deux politiques différentes pour pouvoir les diviser plus tard si vous avez besoin d'un ensemble d'autorisations pour un autre utilisateur. Vous pouvez également regrouper les deux autorisations dans une seule politique, puis associer cette dernière à l'utilisateur IAM nommé Zhang Wei. Vous pouvez aussi attacher une politique à un groupe d'utilisateurs auquel Zhang appartient, ou à un rôle que Zhang peut endosser. Ainsi, lorsque Zhang consulte le contenu d'un compartiment S3, ses demandes sont autorisées. S'il tente de créer un nouvel utilisateur IAM, sa demande est rejetée car il ne dispose pas de l'autorisation appropriée.

Vous pouvez utiliser une limite de permissions sur Zhang pour veiller à ce qu'il n'ait jamais accès au compartiment S3 `DOC-EXAMPLE-BUCKET1`. Pour ce faire, déterminez le nombre maximum d'autorisations dont vous souhaitez que Zhang dispose. Dans ce cas, vous pouvez contrôler ses actions à l'aide de ses politiques d'autorisations. Ici, vous veillez simplement à ce qu'il n'accède pas au compartiment confidentiel. Vous utilisez donc la politique suivante pour définir les limites de Zhang afin d'autoriser toutes les AWS actions pour Amazon S3 et quelques autres services, mais de refuser l'accès au compartiment `DOC-EXAMPLE-BUCKET1` S3. Étant donné que la limite d'autorisations ne permet pas tous les actions IAM, elle empêche Zhang de supprimer sa limite (ou celle d'une autre personne).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PermissionsBoundarySomeServices",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:*",
                "dynamodb:*",
                "ec2:*",
                "s3:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "PermissionsBoundaryNoConfidentialBucket",
```

```
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
            ]
        }
    ]
}
```

Lorsque vous attribuez une telle politique à un utilisateur en tant que limite d'autorisations, n'oubliez pas qu'elle n'accorde aucune autorisation. Elle définit le nombre maximum d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM. Pour plus d'informations sur les limites d'autorisations, consultez Limites d'autorisations pour les entités IAM.

Pour obtenir des informations détaillées sur les procédures mentionnées précédemment, reportez-vous aux ressources suivantes :

- Pour en savoir plus sur la création d'une politique IAM susceptible d'être attachée à un principal, consultez Création de politiques IAM.
- Pour apprendre à attacher une politique IAM à un principal, consultez Ajout et suppression d'autorisations basées sur l'identité IAM.
- Pour consulter un exemple de politique d'octroi d'un accès total à EC2, reportez-vous à Amazon EC2 : autorise l'accès EC2 complet dans une région spécifique, par programmation et dans la console.
- Pour autoriser un accès en lecture seule à un compartiment S3, utilisez les deux premières instructions de l'exemple de politique suivant : Amazon S3 : autorise l'accès en lecture et en écriture aux objets d'un compartiment S3, par programmation et dans la console.
- Pour consulter un exemple de politique permettant aux utilisateurs de définir leurs informations d'identification (leur mot de passe de console, leurs clés d'accès par programmation ou leurs dispositifs MFA, par exemple), veuillez consulter AWS: permet aux utilisateurs IAM authentifiés par MFA de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

Contrôle de l'accès aux identités

Vous pouvez utiliser des politiques IAM pour contrôler ce que vos utilisateurs peuvent faire à une identité en créant une politique que vous attachez à tous les utilisateurs via un groupe d'utilisateurs.

Pour cela, créez une politique qui limite les opérations susceptibles d'être effectuées sur une identité ou les autorisations d'accès.

Par exemple, vous pouvez créer un groupe d'utilisateurs nommé AllUsers, puis l'associer à tous les utilisateurs. Lorsque vous créez le groupe d'utilisateurs, vous pouvez donner à tous vos utilisateurs l'accès à la définition de leurs informations d'identification comme décrit dans la section précédente. Vous pouvez ensuite créer une politique qui interdit l'accès en modification du groupe d'utilisateurs, sauf si le nom d'utilisateur est inclus dans la condition de la politique. Mais cette partie de la politique interdit uniquement l'accès à tous, à l'exception des utilisateurs répertoriés. Vous devez également inclure les autorisations visant à permettre toutes les actions de gestion de groupe d'utilisateurs pour toutes les personnes du groupe. Enfin, vous attachez cette politique au groupe d'utilisateurs de façon à ce qu'elle soit appliquée à tous les utilisateurs. De cette façon, lorsqu'un utilisateur non spécifié dans la politique tente d'apporter des modifications au groupe d'utilisateurs, la demande est refusée.

Pour créer cette politique avec l'éditeur visuel

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, sélectionnez Policies (Politiques).

   Si vous sélectionnez Politiques pour la première fois, la page Bienvenue dans les politiques gérées s'affiche. Sélectionnez Get started (Mise en route).
3. Sélectionnez Créer une politique.
4. Dans la section Éditeur de politiques, choisissez l'option Visuel.
5. Dans Sélectionner un service, choisissez IAM.
6. Dans Actions autorisées, tapez **`group`** dans la zone de recherche. L'éditeur visuel affiche toutes les actions IAM qui contiennent le mot `group`. Activez toutes les cases à cocher.
7. Choisissez Ressources pour spécifier les ressources de votre politique. En fonction des actions que vous avez choisies, vous devriez voir les types de ressources groupe et utilisateur.

   - groupe : choisissez Ajouter des ARN. Pour Ressource dans, sélectionnez l'option Tout compte. Cochez la case Tout nom de groupe avec chemin, puis tapez le nom du groupe d'utilisateurs **`AllUsers`**. Puis, choisissez Ajouter des ARN.
   - utilisateur : cochez la case en regard de Toute personne dans ce compte.

   Une des actions que vous avez choisies, `ListGroups`, ne prend pas en charge l'utilisation de ressources spécifiques. Vous n'avez pas besoin de choisir Toutes les ressources pour cette

action. Lorsque vous enregistrez votre politique ou que vous l'affichez dans l'éditeur JSON, vous pouvez voir que IAM crée automatiquement un nouveau bloc d'autorisation qui accorde à cette action une autorisation sur toutes les ressources.

8. Pour ajouter un autre bloc d'autorisation, choisissez Ajouter d'autres autorisations.
9. Choisissez Sélectionner un service, puis IAM.
10. Choisissez Actions autorisées, puis Basculer pour refuser les autorisations. Si vous procédez ainsi, l'ensemble du bloc est utilisé pour refuser des autorisations.
11. Dans la zone de recherche, saisissez **`group`**. L'éditeur visuel affiche toutes les actions IAM qui contiennent le mot `group`. Activez les cases à cocher en regard des actions suivantes :

    - CreateGroup
    - DeleteGroup
    - RemoveUserFromGroup
    - AttachGroupPolicy
    - DeleteGroupPolicy
    - DetachGroupPolicy
    - PutGroupPolicy
    - UpdateGroup
12. Choisissez Ressources pour spécifier les ressources de votre politique. En fonction des actions que vous avez choisies, vous devriez voir le type de ressource group. Choisissez Ajouter des ARN. Pour Ressource dans, sélectionnez l'option Tout compte. Pour Tout nom de groupe avec chemin, tapez le nom du groupe d'utilisateurs **`AllUsers`**. Puis, choisissez Ajouter des ARN.
13. Choisissez Demander des conditions – facultatif, puis Ajouter une autre condition. Remplissez le formulaire avec les valeurs suivantes :

    - Clé de condition : choisissez aws:username
    - Qualifier (Qualificateur) : sélectionnez Default (Valeur par défaut)
    - Opérateur — Choisissez StringNotEquals
    - Valeur : tapez **`srodriguez`**, puis choisissez Ajouter pour ajouter une autre valeur. Tapez **`mjackson`**, puis choisissez Ajouter pour ajouter une autre valeur. Tapez **`adesai`**, puis choisissez Ajouter une condition.

    Grâce à cette condition, l'accès est refusé aux actions spécifiées de gestion du groupe d'utilisateurs lorsque l'utilisateur à l'origine de l'action n'est pas inclus dans la liste. Dans la

mesure où cette option refuse explicitement l'autorisation, elle remplace le bloc précédent qui a autorisé les utilisateurs à demander les actions. Les utilisateurs de la liste ne se voient pas refuser l'accès et ils sont autorisés dans le premier bloc d'autorisation, afin qu'ils puissent entièrement gérer le groupe d'utilisateurs.

14. Lorsque vous avez terminé, choisissez Next.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'option éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez Restructuration de politique.

15. Sur la page Vérifier et créer, pour le Nom de la politique, tapez **`LimitAllUserGroupManagement`**. Pour la Description, saisissez **`Allows all users read-only access to a specific user group, and allows only specific users access to make changes to the user group`**. Vérifiez les Autorisations définies dans cette politique pour vous assurer que vous les avez accordées comme prévu. Ensuite, choisissez Créer une politique pour enregistrer votre nouvelle politique.
16. Attachez la politique à votre groupe d'utilisateurs. Pour plus d'informations, veuillez consulter Ajout et suppression d'autorisations basées sur l'identité IAM.

Vous pouvez aussi créer la même politique à l'aide de cet exemple de document de politique JSON. Pour afficher cette politique JSON, veuillez consulter IAM : autorise des utilisateurs spécifiques à gérer un groupe par programmation et dans la console. Pour obtenir des instructions détaillées sur la création d'une politique à l'aide d'un document JSON, veuillez consulter the section called “Création de politiques à l'aide de l'éditeur JSON”.

Contrôle de l'accès aux politiques

Vous pouvez contrôler la manière dont vos utilisateurs peuvent appliquer les politiques AWS gérées. Pour cela, attachez cette politique à tous vos utilisateurs. Idéalement, vous pouvez le faire en utilisant un groupe d'utilisateurs.

Par exemple, vous pouvez créer une politique qui permet aux utilisateurs d'associer uniquement l'IAM UserChangePassword et les politiques PowerUserAccess AWS gérées à un nouvel utilisateur, groupe d'utilisateurs ou rôle IAM.

Pour les politiques gérées par les clients, vous pouvez contrôler qui peut les créer, les mettre à jour et les supprimer. Vous pouvez contrôler qui peut attacher et détacher les politiques vers et depuis des entités du principal (groupes d'utilisateurs, utilisateurs et rôles). Vous pouvez également contrôler quelles politiques peuvent être attachées ou détachées de ces entités.

Par exemple, vous pouvez autoriser un administrateur de compte à créer, mettre à jour et supprimer des politiques. Ensuite, vous accordez des autorisations à un responsable d'équipe ou un autre administrateur disposant de droits limités afin de lui permettre d'attacher et de détacher ces politiques des entités du principal gérées par l'administrateur doté de droits limités.

Pour de plus amples informations, veuillez consulter les ressources suivantes :

- Pour en savoir plus sur la création d'une politique IAM susceptible d'être attachée à un principal, consultez Création de politiques IAM.
- Pour apprendre à attacher une politique IAM à un principal, consultez Ajout et suppression d'autorisations basées sur l'identité IAM.
- Pour consulter un exemple de politique permettant de limiter l'utilisation des politiques gérées, reportez-vous à IAM : limite les politiques gérées pouvant être appliquées à un utilisateur, groupe ou rôle IAM.

Contrôle des autorisations pour la création, mise à jour et suppression de politiques gérées par le client

Vous pouvez utiliser des politiques IAM pour contrôler qui est autorisé à créer, mettre à jour et supprimer les politiques gérées par le client dans votre Compte AWS. La liste suivante répertorie les opérations d'API ayant un lien direct avec la création, la mise à jour, la suppression et la gestion des politiques ou versions de politiques :

- CreatePolicy
- CreatePolicyVersion
- DeletePolicy
- DeletePolicyVersion
- SetDefaultPolicyVersion

Les opérations d'API ci-dessus correspondent à des actions que vous pouvez autoriser ou refuser autrement dit, des autorisations que vous pouvez accorder à l'aide d'une politique IAM.

Examinons l'exemple de politique suivant. Il autorise un utilisateur à créer, mettre à jour (autrement dit, créer une version de politique), supprimer et définir une version par défaut pour toutes les politiques gérées par le client dans l' Compte AWS. Cette politique permet également à l'utilisateur d'obtenir des politiques ou d'en afficher la liste. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

Example Exemple de politique qui autorise la création, mise à jour, suppression, obtention et définition de la version par défaut pour toutes les politiques

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:CreatePolicy",
      "iam:CreatePolicyVersion",
      "iam:DeletePolicy",
      "iam:DeletePolicyVersion",
      "iam:GetPolicy",
      "iam:GetPolicyVersion",
      "iam:ListPolicies",
      "iam:ListPolicyVersions",
      "iam:SetDefaultPolicyVersion"
    ],
    "Resource": "*"
  }
}
```

Vous pouvez créer des politiques qui limitent l'utilisation de ces opérations d'API aux seules politiques gérées que vous spécifiez. Par exemple, il est possible d'autoriser un utilisateur à définir la version par défaut et supprimer des versions de politique, mais uniquement pour certaines politiques gérées par le client. Pour ce faire, vous spécifiez l'ARN de la politique dans l'élément `Resource` de la politique qui accorde les autorisations.

L'exemple suivant illustre une politique qui autorise un utilisateur à supprimer des versions de politique et à définir la version par défaut. Mais ces actions sont uniquement autorisées pour les politiques gérées par le client comportant le chemin/TEAM-A/. L'ARN de la politique gérée par le client est spécifié dans l'élément `Resource` de la politique. Dans cet exemple, l'ARN inclut un chemin d'accès et un caractère générique et, par conséquent, correspond à toutes les politiques gérées

par le client qui incluent le chemin /TEAM-A/. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

Pour de plus amples informations sur l'utilisation de chemins dans les noms de politiques gérées par le client, veuillez consulter Noms conviviaux et chemins.

Example Exemple de politique qui autorise la suppression de versions de politiques et la définition de la version par défaut pour des politiques spécifiques uniquement

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:DeletePolicyVersion",
      "iam:SetDefaultPolicyVersion"
    ],
    "Resource": "arn:aws:iam::account-id:policy/TEAM-A/*"
  }
}
```

Contrôle des autorisations pour attacher et détacher des politiques gérées

Vous pouvez également utiliser des politiques IAM pour n'autoriser les utilisateurs qu'à utiliser des politiques gérées spécifiques. En effet, vous pouvez contrôler les autorisations qu'un utilisateur est autorisé à accorder à d'autres entités du principal.

La liste suivante répertorie les opérations d'API utilisées pour attacher et détacher des politiques gérées d'entités du principal :

- AttachGroupPolicy
- AttachRolePolicy
- AttachUserPolicy
- DetachGroupPolicy
- DetachRolePolicy
- DetachUserPolicy

Vous pouvez créer des politiques qui limitent l'utilisation de ces opérations d'API aux seules politiques gérées et/ou entités de principal que vous spécifiez. Par exemple, il est possible d'autoriser un utilisateur à attacher des politiques gérées, mais uniquement celles que vous spécifiez. Ou, il est possible d'autoriser un utilisateur à attacher des politiques gérées, mais uniquement aux entités du principal que vous spécifiez.

L'exemple suivant illustre une politique qui autorise un utilisateur à attacher des politiques gérées aux groupes d'utilisateurs et rôles comportant le chemin /TEAM-A/ uniquement. Les ARN de groupe d'utilisateurs et de rôle sont spécifiés dans l'élément de politique `Resource`. Dans cet exemple, les ARN incluent un chemin d'accès et un caractère générique et, par conséquent, correspondent à tous les groupes d'utilisateurs et rôles qui incluent le chemin /TEAM-A/. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

Example Exemple de politique qui autorise l'utilisateur à attacher des politiques gérées à des groupes d'utilisateurs ou rôles spécifiques uniquement

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:AttachGroupPolicy",
      "iam:AttachRolePolicy"
    ],
    "Resource": [
      "arn:aws:iam::account-id:group/TEAM-A/*",
      "arn:aws:iam::account-id:role/TEAM-A/*"
    ]
  }
}
```

Vous pouvez limiter encore davantage les actions de l'exemple précédent, de manière à n'affecter que des politiques spécifiques. En d'autres termes, vous pouvez contrôler les autorisations qu'un utilisateur est autorisé à attacher à d'autres entités du principal, en ajoutant une condition à la politique.

Dans l'exemple suivant, la condition garantit que les autorisations `AttachGroupPolicy` et `AttachRolePolicy` sont octroyées uniquement si la politique devant être attachée correspond à l'une des politiques spécifiées. La condition utilise la clé de condition `iam:PolicyARN` pour

déterminer la ou les politiques qui peuvent être attachées. L'exemple de politique suivant développe l'exemple précédent. Il autorise un utilisateur à attacher uniquement les politiques gérées comportant le chemin /TEAM-A/ aux seuls groupes d'utilisateurs et rôles incluent le chemin /TEAM-A/. Pour apprendre à créer une politique à l'aide de cet exemple de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:AttachGroupPolicy",
      "iam:AttachRolePolicy"
    ],
    "Resource": [
      "arn:aws:iam::account-id:group/TEAM-A/*",
      "arn:aws:iam::account-id:role/TEAM-A/*"
    ],
    "Condition": {"ArnLike":
      {"iam:PolicyARN": "arn:aws:iam::account-id:policy/TEAM-A/*"}
    }
  }
}
```

Cette politique utilise l'opérateur de condition `ArnLike`, mais vous pouvez également utiliser l'opérateur de condition `ArnEquals` car ces deux opérateurs de condition se comportent de manière identique. Pour de plus amples informations sur `ArnLike` et `ArnEquals`, veuillez consulter Opérateurs de condition d'Amazon Resource Name (ARN) dans la section Types de conditions des Références des éléments de politique.

Par exemple, vous pouvez limiter l'utilisation de ces actions aux politiques gérées que vous spécifiez. Pour ce faire, vous spécifiez l'ARN de la politique dans l'élément `Condition` de la politique qui accorde les autorisations. Par exemple, pour spécifier l'ARN d'une politique gérée par le client :

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::123456789012:policy/POLICY-NAME"}
}
```

Vous pouvez également spécifier l'ARN d'une stratégie AWS gérée dans l'`Condition`élément d'une stratégie. L'ARN d'une politique AWS gérée utilise l'alias spécial contenu `aws` dans l'ARN de la politique au lieu d'un identifiant de compte, comme dans cet exemple :

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::aws:policy/AmazonEC2FullAccess"}
}
```

Contrôle de l'accès aux ressources

Vous pouvez contrôler l'accès aux ressources à l'aide d'une politique basée sur les identités ou sur les ressources. Dans le cadre d'une politique basée sur les identités, vous attachez la politique à une identité et vous spécifiez à quelles ressources cette identité peut accéder. Dans le cadre d'une politique basée sur les ressources, vous attachez une politique à la ressource que vous souhaitez contrôler. Dans la politique, vous spécifiez quels principaux peuvent accéder à cette ressource. Pour de plus amples informations sur les deux types de stratégie, veuillez consulter Politiques basées sur l'identité et Politiques basées sur une ressource.

Pour de plus amples informations, veuillez consulter les ressources suivantes :

- Pour en savoir plus sur la création d'une politique IAM susceptible d'être attachée à un principal, consultez Création de politiques IAM.
- Pour apprendre à attacher une politique IAM à un principal, consultez Ajout et suppression d'autorisations basées sur l'identité IAM.
- Amazon S3 prend en charge l'utilisation des politiques basées sur les ressources sur leurs compartiments. Pour de plus amples informations, veuillez consulter Exemples de politique de compartiment.

Des créateurs de ressource ne reçoivent pas automatiquement des autorisations

Si vous vous connectez à l'aide des Utilisateur racine d'un compte AWS informations d'identification, vous êtes autorisé à effectuer toute action sur les ressources appartenant au compte. Toutefois, ce n'est pas le cas pour les utilisateurs IAM. Un utilisateur IAM peut recevoir un accès pour créer une ressource, mais les autorisations de cet utilisateur, même pour cette ressource, se limitent à ce qui a été explicitement accordé. Cela signifie que si vous créez une ressource, par exemple un rôle IAM, vous n'avez pas automatiquement l'autorisation de modifier ou de supprimer ce rôle. De plus, votre autorisation peut être révoquée à tout moment par le propriétaire du compte ou par un autre utilisateur disposant d'un accès pour gérer vos autorisations.

Contrôle de l'accès aux principaux dans un compte spécifique

Vous pouvez accorder directement à des utilisateurs IAM de votre propre compte un accès à vos ressources. Si des utilisateurs d'un autre compte doivent accéder à vos ressources, vous pouvez créer un rôle IAM. Un rôle est une entité incluant des autorisations, mais qui n'est pas associée à un utilisateur spécifique. Les utilisateurs d'autres comptes peuvent ensuite endosser le rôle et accéder aux ressources en fonction des autorisations que vous avez attribuées à ce rôle. Pour plus d'informations, veuillez consulter Fournir l'accès à un utilisateur IAM dans un autre utilisateur Compte AWS dont vous êtes le propriétaire.

> **Note**
>
> Certains services prennent en charge les politiques basées sur les ressources, comme décrit dansPolitiques basées sur l'identité et Politiques basées sur une ressource (Amazon S3, Amazon SNS et Amazon SQS, par ex.). Pour ces services, une alternative à l'utilisation de rôles consiste à attacher une politique à la ressource (compartiment, rubrique ou file d'attente) que vous voulez partager. La politique basée sur les ressources peut spécifier le AWS compte autorisé à accéder à la ressource.

Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises

Utilisez les informations de la section suivante pour contrôler qui peut accéder à vos utilisateurs et rôles IAM, ainsi que les ressources auxquelles vos utilisateurs et rôles peuvent accéder. Pour des informations plus générales et des exemples de contrôle de l'accès à d'autres AWS ressources, y compris à d'autres ressources IAM, consultezBalisage des ressources IAM.

> **Note**
>
> Pour plus d'informations sur la sensibilité à la casse des clés de balises et les valeurs de clés de balises, consultez Case sensitivity (français non garanti).

Les balises peuvent être attachées à la ressource IAM, transmises dans la demande ou attachées au principal qui effectue la demande. Un utilisateur ou rôle IAM peut être à la fois une ressource et un principal. Par exemple, vous pouvez écrire une politique qui permet à un utilisateur de répertorier

les groupes d'un utilisateur. Cette opération est autorisée uniquement si l'utilisateur effectuant la demande (le principal) a la même balise `project=blue` que l'utilisateur qu'il essaie de consulter. Dans cet exemple, l'utilisateur peut afficher l'appartenance au groupe de n'importe quel utilisateur, y compris lui-même, tant qu'il travaille sur le même projet.

Pour contrôler l'accès basé sur des balises, vous devez fournir les informations des balises dans l'élément de condition d'une politique. Lorsque vous créez une politique IAM, vous pouvez utiliser des balises IAM et la clé de condition de balise associée pour contrôler l'accès à chacun des éléments suivants :

- Resource (Ressource) : contrôlez l'accès aux ressources d'utilisateur ou de rôle en fonction de leurs balises. Pour ce faire, utilisez la clé de condition aws :ResourceTag/***key-name*** pour spécifier quelle paire clé-valeur de balise doit être attachée à la ressource. Pour plus d'informations, consultez Contrôle de l'accès aux ressources AWS.
- Request (Demande) : contrôlez les balises pouvant être transmises dans une demande IAM. Pour ce faire, utilisez la clé de condition aws :RequestTag/***key-name*** pour spécifier les balises qui peuvent être ajoutées, modifiées ou supprimées pour un utilisateur ou un rôle IAM. Cette clé est utilisée de la même manière pour les ressources IAM et les autres AWS ressources. Pour plus d'informations, consultez Contrôle de l'accès au cours des demandes AWS.
- Principal : contrôlez ce que la personne à l'origine de la demande (le principal) est autorisée à faire en fonction des balises qui sont attachées à l'utilisateur ou au rôle IAM de cette personne. Pour ce faire, utilisez la clé de condition aws :PrincipalTag/***key-name*** pour spécifier les balises qui doivent être associées à l'utilisateur ou au rôle IAM avant que la demande ne soit autorisée.
- N'importe quelle partie du processus d'autorisation : utilisez la clé de TagKeys condition aws : pour contrôler si des clés de balise spécifiques peuvent être utilisées dans une demande ou par un principal. Dans ce cas, la valeur de la clé n'importe pas. Cette clé se comporte de la même manière pour IAM et les autres AWS services. Toutefois, lorsque vous balisez un utilisateur dans IAM, cette action contrôle également si le principal peut adresser la demande à n'importe quel service. Pour plus d'informations, veuillez consulter Contrôle de l'accès en fonction des clés de balise.

Vous pouvez créer une politique IAM à l'aide de l'éditeur visuel, à l'aide de JSON ou en important une politique gérée existante. Pour plus de détails, consultez Création de politiques IAM.

> **Note**
>
> Vous pouvez également transmettre des balises de session lorsque vous endossez un rôle IAM ou fédérez un utilisateur. Celles-ci ne sont valides que pendant la durée de la session.

Contrôle de l'accès pour les principaux IAM

Vous pouvez contrôler ce que le principal est autorisé à faire en fonction des balises attachées à l'identité de cette personne.

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui permet à tout utilisateur de ce compte d'afficher l'appartenance au groupe de n'importe quel utilisateur, y compris lui-même, tant qu'il travaille sur le même projet. Cette opération est autorisée uniquement lorsque la balise de ressource de l'utilisateur et la balise du principal ont la même valeur pour la clé de balise `project`. Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "iam:ListGroupsForUser",
            "Resource": "arn:aws:iam::111222333444:user/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/project":
 "${aws:PrincipalTag/project}"}
            }
        }]
}
```

Contrôle de l'accès en fonction des clés de balise

Vous pouvez utiliser des balises dans vos politiques IAM pour contrôler si des clés de balise spécifiques peuvent être utilisées dans une demande ou par un principal.

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise la suppression de la balise avec la clé `temporary` uniquement, des utilisateurs. Pour utiliser cette

politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:UntagUser",
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": ["temporary"]}}
    }]
}
```

Contrôle de l'accès aux AWS ressources à l'aide de balises

Vous pouvez utiliser des balises pour contrôler l'accès à vos AWS ressources qui prennent en charge le balisage, y compris les ressources IAM. Vous pouvez baliser des utilisateurs et des rôles IAM pour contrôler les éléments auxquels ils peuvent accéder. Pour savoir comment baliser des utilisateurs et des rôles IAM, consultez Balisage des ressources IAM. En outre, vous pouvez contrôler l'accès aux ressources IAM suivantes : politiques gérées par le client, fournisseurs d'identité IAM, profils d'instance, certificats de serveur et dispositifs MFA virtuels. Pour accéder à un didacticiel sur la création et le test d'une politique permettant aux rôles IAM avec des balises principaux d'accéder aux ressources avec des balises correspondantes, veuillez consulter Tutoriel IAM : définir les autorisations d'accès aux AWS ressources en fonction des balises. Utilisez les informations de la section suivante pour contrôler l'accès à d'autres AWS ressources, y compris les ressources IAM, sans baliser les utilisateurs ou les rôles IAM.

Avant d'utiliser des balises pour contrôler l'accès à vos AWS ressources, vous devez comprendre comment AWS octroyer l'accès. AWS est composé de collections de ressources. Une instance Amazon EC2 est une ressource. Un compartiment Amazon S3 est une ressource. Vous pouvez utiliser l' AWS API AWS CLI, le ou le AWS Management Console pour effectuer une opération, telle que la création d'un compartiment dans Amazon S3. Dans ce cas, vous envoyez une demande pour cette opération. Votre demande spécifie une action, une ressource, une entité mandataire (utilisateur ou rôle), un compte principal et toutes les informations nécessaires relatives à la demande. Toutes ces informations fournissent le contexte.

AWS vérifie ensuite que vous (l'entité principale) êtes authentifié (connecté) et autorisé (autorisé) à effectuer l'action spécifiée sur la ressource spécifiée. Lors de l'autorisation, AWS vérifie toutes

les politiques applicables au contexte de votre demande. La plupart des politiques sont stockées AWS sous forme de documents JSON et spécifient les autorisations pour les entités principales. Pour plus d'informations sur les types de politiques et les utilisations, veuillez consulter Politiques et autorisations dans IAM.

AWS autorise la demande uniquement si chaque partie de votre demande est autorisée par les politiques. Pour afficher un schéma et en savoir plus sur l'infrastructure IAM, consultez Fonctionnement de IAM. Pour obtenir des détails sur la façon dont IAM détermine si une demande est autorisée, reportez-vous à Logique d'évaluation de politiques.

Les balises sont une autre considération concernant ce processus, car elles peuvent être attachées à la ressource ou transmises dans la demande aux services qui prennent en charge le balisage. Pour contrôler l'accès basé sur des balises, vous devez fournir les informations des balises dans l'élément de condition d'une politique. Pour savoir si un AWS service prend en charge le contrôle d'accès à l'aide de balises, consultez AWS services qui fonctionnent avec IAM et recherchez les services dont la valeur est « Oui » dans la colonne ABAC. Choisissez le nom du service pour afficher la documentation sur l'autorisation et le contrôle d'accès de ce service.

Vous pouvez ensuite créer une politique IAM qui autorise ou refuse l'accès à une ressource en fonction de la balise de cette ressource. Dans cette politique , vous pouvez utiliser des clés de condition de balise pour contrôler l'accès à l'un des éléments suivants :

- Ressource : contrôlez l'accès aux ressources de AWS service en fonction des balises figurant sur ces ressources. Pour ce faire, utilisez la clé de condition ResourceTag/***key-name*** pour déterminer s'il faut autoriser l'accès à la ressource en fonction des balises associées à la ressource.
- Request (Demande) : contrôle les balises qui peuvent être transmises dans une demande. Pour ce faire, utilisez la clé de condition aws :RequestTag/***key-name*** pour spécifier les paires clé-valeur de balise qui peuvent être transmises dans une demande de balisage d'une ressource. AWS
- N'importe quelle partie du processus d'autorisation : utilisez la clé de TagKeys condition aws : pour contrôler si des clés de balise spécifiques peuvent figurer dans une demande.

Vous pouvez créer une politique IAM visuellement, à l'aide de JSON, ou en important une politique gérée existante. Pour plus de détails, consultez Création de politiques IAM.

> **Note**
>
> Certains services permettent aux utilisateurs d'attribuer des balises au moment de la création de la ressource (s'ils ont les autorisations d'utiliser l'action qui crée la ressource).

Contrôle de l'accès aux ressources AWS

Vous pouvez utiliser des conditions dans vos politiques IAM pour contrôler l'accès aux AWS ressources en fonction des balises associées à ces ressources. Pour ce faire, vous pouvez utiliser la clé de condition globale `aws:ResourceTag/`*`tag-key`* ou une clé spécifique au service. Certains services prennent en charge uniquement la version spécifique au service de cette clé et non la version globale.

> **Warning**
>
> N'essayez pas de contrôler les personnes susceptibles de faire passer un rôle en étiquetant ce rôle, puis en utilisant la clé de condition `ResourceTag` dans une politique avec l'action `iam:PassRole`. Cette approche ne produit pas des résultats fiables. Pour de plus amples informations sur les autorisations requises pour transmettre un rôle à un service, veuillez consulter Octroi d'autorisations à un utilisateur pour transférer un rôle à un service AWS.

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise le démarrage ou l'arrêt d'instances Amazon EC2. Ces opérations sont autorisées uniquement si la balise d'instance `Owner` a la valeur du nom d'utilisateur. Cette politique définit des autorisations pour l'accès à la console et par programmation.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Owner": "${aws:username}"}
```

```
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

Vous pouvez rattacher cette politique aux utilisateurs IAM de votre compte. Si un utilisateur nommé `richard-roe` tente de démarrer une instance Amazon EC2, l'instance doit être balisée `0wner=richard-roe` ou `owner=richard-roe`. Dans le cas contraire, il se verra refuser l'accès. La clé de balise `0wner` correspond à la fois à `0wner` et `owner`, car les noms de clé de condition ne sont pas sensibles à la casse. Pour plus d'informations, consultez Éléments de politique JSON IAM : Condition.

Cet exemple montre comment créer une politique basée sur l'identité qui utilise la balise de principal `team` dans l'ARN de la ressource. Cette politique autorise la suppression des files d'attente Amazon Simple Queue Service, mais uniquement si le nom de la file d'attente commence par le nom de l'équipe suivi de `-queue`. Par exemple, `qa-queue` si `qa` est le nom de l'équipe pour la balise de principal `team`.

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Sid": "AllQueueActions",
        "Effect": "Allow",
        "Action": "sqs:DeleteQueue",
        "Resource": "arn:aws:sqs:us-east-2::${aws:PrincipalTag/team}-queue"
      }
}
```

Contrôle de l'accès au cours des demandes AWS

Vous pouvez utiliser des conditions dans vos politiques IAM pour contrôler les paires clé-valeur de balise qui peuvent être transmises dans une demande qui applique des balises à une ressource. AWS

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'utilisation de l'action `CreateTags` d'Amazon EC2 pour attacher des balises à une instance. Vous

pouvez attacher des balises uniquement si la balise contient la clé `environment` et les valeurs `production` ou `preprod`. En tant que bonne pratique, utilisez le modificateur `ForAllValues` avec la clé de condition `aws:TagKeys` pour indiquer que seule la clé `environment` est autorisée dans la demande. Cela empêche les utilisateurs d'inclure d'autres clés, notamment en utilisant accidentellement `Environment` au lieu de `environment`.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "ec2:CreateTags",
        "Resource": "arn:aws:ec2:*:*:instance/*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/environment": [
                    "preprod",
                    "production"
                ]
            },
            "ForAllValues:StringEquals": {"aws:TagKeys": "environment"}
        }
    }
}
```

Contrôle de l'accès en fonction des clés de balise

Vous pouvez utiliser une condition dans vos politiques IAM pour contrôler si des clés de balise spécifiques peuvent être utilisées dans une requête.

Lorsque vous utilisez des politiques pour contrôler l'accès à l'aide de balises, nous vous recommandons d'utiliser la clé de `aws:TagKeys` condition. AWS les services qui prennent en charge les balises peuvent vous permettre de créer plusieurs noms de clés de balise qui ne diffèrent qu'au cas par cas, par exemple en balisant une instance `stack=production` Amazon EC2 avec et. `Stack=test` Les noms de clé ne sont pas sensibles à la casse dans les conditions des politiques. Cela signifie que si vous spécifiez `"aws:ResourceTag/TagKey1": "Value1"` dans l'élément de condition de votre politique, la condition correspond à une clé de balise de ressource nommée `TagKey1` ou `tagkey1`, mais pas aux deux. Pour éviter la duplication de balises avec une clé qui ne varie que par la casse, utilisez la condition `aws:TagKeys` pour définir les clés de balise que vos utilisateurs peuvent appliquer, ou utilisez les politiques de balises, disponibles avec AWS

Organizations. Pour de plus d'informations, veuillez consulter Politiques de balises dans le Guide de l'utilisateur Organizations.

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise la création et la balise d'un secret Secrets Manager, mais uniquement avec les clés de balise `environment` ou `cost-center`. La condition `Null` garantit que la condition a la valeur `false` s'il n'y a pas de balises dans la demande.

```
{
        "Effect": "Allow",
        "Action": [
            "secretsmanager:CreateSecret",
            "secretsmanager:TagResource"
        ],
        "Resource": "*",
        "Condition": {
            "Null": {
                "aws:TagKeys": "false"
            },
            "ForAllValues:StringEquals": {
                "aws:TagKeys": [
                    "environment",
                    "cost-center"
                ]
            }
        }
}
```

Accès intercompte aux ressources dans IAM

Pour certains AWS services, vous pouvez accorder un accès multicompte à vos ressources à l'aide d'IAM. Pour cela, vous pouvez attacher une politique de ressources directement à la ressource que vous voulez partager, ou bien utiliser un rôle en tant que proxy.

Pour partager directement une ressource, la ressource que vous voulez partager doit prendre en charge les politiques basées sur les ressources. Contrairement à une politique basée sur l'identité pour un rôle, une politique basée sur une ressource spécifie qui (quel principal) peut accéder à cette ressource.

Utilisez un rôle en tant que proxy lorsque vous voulez accéder aux ressources d'un autre compte qui ne prend pas en charge les politiques basées sur les ressources.

Pour en savoir plus sur les différences entre ces types de politiques, consultez Politiques basées sur l'identité et Politiques basées sur une ressource.

> **Note**
>
> Les politiques IAM basées sur les ressources et les rôles ne délèguent l'accès entre les comptes qu'au sein d'une seule partition. Par exemple, vous avez un compte dans la région USA Ouest (Californie du Nord) dans la partition `aws` standard. Vous avez également un compte dans la région Chine dans la partition `aws-cn`. Vous ne pouvez pas utiliser une politique basée sur les ressources dans votre compte en Chine pour autoriser l'accès aux utilisateurs de votre compte standard AWS .

Accès intercompte à l'aide de rôles

Les politiques basées sur les ressources ne sont pas prises en charge par tous les AWS services. Pour ces services, vous pouvez utiliser des rôles IAM intercomptes afin de centraliser la gestion des autorisations lorsque vous fournissez un accès intercompte à plusieurs services. Un rôle IAM entre comptes est un rôle IAM qui inclut une politique de confiance qui permet aux principaux IAM d'un autre AWS compte d'assumer ce rôle. En termes simples, vous pouvez créer un rôle dans un AWS compte qui délègue des autorisations spécifiques à un autre AWS compte.

Pour de plus amples informations sur l'attachement d'une politique à une identité IAM, veuillez consulter Gestion des politiques IAM.

> **Note**
>
> Lorsqu'un principal passe à un rôle pour utiliser temporairement les autorisations de ce rôle, il abandonne ses autorisations d'origine et reprend les autorisations attribuées au rôle qu'il a assumé.

Examinons le processus global tel qu'il s'applique au logiciel du partenaire APN qui doit accéder à un compte client.

1. Le client crée un rôle IAM dans son propre compte avec une politique qui autorise l'accès aux ressources Amazon S3 dont le partenaire APN a besoin. Dans cet exemple, le nom du rôle est `APNPartner`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name"
            ]
        }
    ]
}
```

2. Le client précise ensuite que le rôle peut être assumé par le AWS compte du partenaire en fournissant l' Compte AWS identifiant du partenaire APN dans la politique de confiance relative au APN`Partner` rôle.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::APN-account-ID:role/APN-user-name"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

3. Le client donne l'Amazon Resource Name (ARN) du rôle au partenaire APN. L'ARN est le nom entièrement qualifié du rôle.

```
arn:aws:iam::APN-ACCOUNT-ID:role/APNPartner
```

> **Note**
>
> Nous vous recommandons d'utiliser un ID externe dans les situations à locataires multiples. Pour plus de détails, consultez Comment utiliser un identifiant externe lorsque vous accordez l'accès à vos AWS ressources à un tiers.

4. Lorsque le logiciel du partenaire APN doit accéder au compte du client, le logiciel appelle l'AssumeRoleAPI AWS Security Token Service avec l'ARN du rôle dans le compte du client. STS renvoie un AWS identifiant temporaire qui permet au logiciel de faire son travail.

Pour un autre exemple de l'octroi d'un accès intercompte à l'aide de rôles, consultez Fournir l'accès à un utilisateur IAM dans un autre utilisateur Compte AWS dont vous êtes le propriétaire. Vous pouvez également suivre le tutoriel Didacticiel IAM : déléguer l'accès entre des comptes AWS à l'aide des rôles IAM.

Accès intercompte à l'aide de politiques basées sur les ressources

Lorsqu'un compte accède à une ressource par le biais d'un autre compte à l'aide d'une politique basée sur les ressources, le principal continue d'utiliser le compte approuvé sans devoir renoncer à ses autorisations au profit des autorisations du rôle. Autrement dit, le principal continue d'avoir accès aux ressources du compte approuvé tout en ayant accès à la ressource du compte d'approbation. Cela est utile pour les tâches de copie d'informations de ou vers la ressources partagée de l'autre compte.

Les principes que vous pouvez spécifier dans une politique basée sur les ressources incluent les comptes, les utilisateurs IAM, les utilisateurs fédérés, les rôles IAM, les sessions de rôle assumé ou les services. AWS Pour plus d'informations, consultez Spécification d'un principal.

Pour savoir si les principaux des comptes situés en dehors de votre zone d'approbation (organisation ou compte d'approbation) peuvent accéder à vos rôles et les assumer, consultez Identification des ressources partagées avec une entité externe.

La liste suivante inclut certains des AWS services qui prennent en charge les politiques basées sur les ressources. Pour obtenir une liste complète du nombre croissant de AWS services qui permettent d'associer des politiques d'autorisation aux ressources plutôt qu'aux principaux, consultez AWS services qui fonctionnent avec IAM et recherchez les services dont la valeur est Oui dans la colonne Basé sur les ressources.

- Compartiments Amazon S3 : la politique est attachée au compartiment, mais la politique contrôle l'accès au compartiment et aux objets qu'il contient. Pour plus d'informations, veuillez consulter contrôle d'accès dans le guide de l'utilisateur du service de stockage simple Amazon. Dans certains cas, il peut être préférable d'utiliser des rôles pour l'accès entre comptes à Amazon S3. Pour plus d'informations, veuillez consulter les exemples de démonstration dans le guide de l'utilisateur du service de stockage simple Amazon.
- Rubriques Amazon Simple Notification Service (Amazon SNS) : pour plus d'informations, consultez Cas d'exemple pour le contrôle d'accès Amazon SNS dans le Guide du développeur Amazon Simple Notification Service.
- Files d'attente Amazon Simple Queue Service (Amazon SQS) : pour de plus amples informations, veuillez consulter Annexe : langage de la politique d'accès dans le Manuel du développeur Amazon Simple Queue Service.

Délégation d' AWS autorisations dans une politique basée sur les ressources

Si une ressource accorde des autorisations aux principaux de votre compte, vous pouvez alors déléguer ces autorisations à des identités IAM spécifiques. Les identités sont des utilisateurs, des groupes d'utilisateurs ou des rôles de votre compte. Vous déléguez des autorisations en attachant une politique à l'identité. Vous pouvez accorder autant d'autorisations que le compte propriétaire de la ressource le permet.

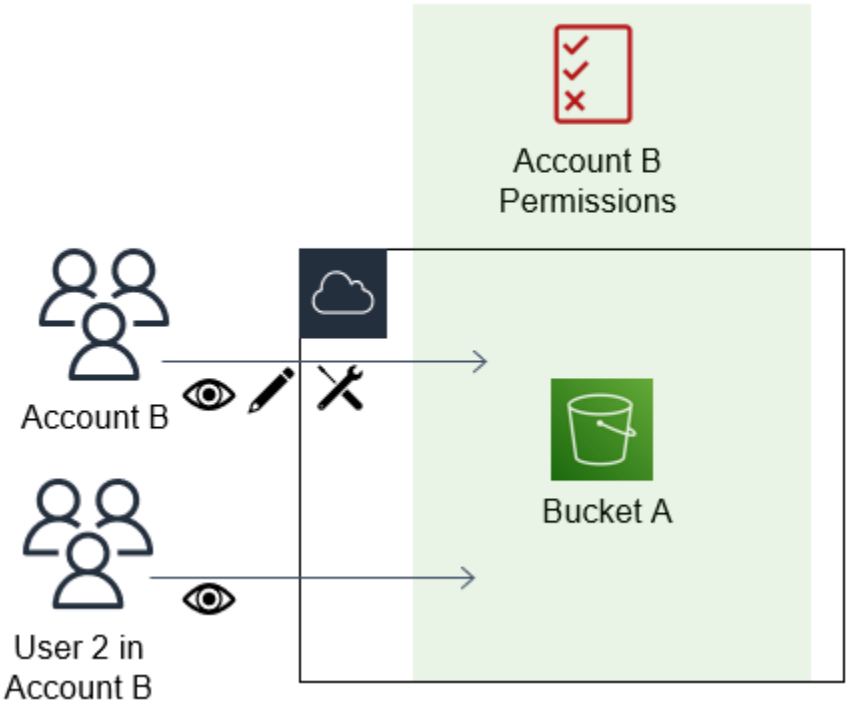
> **Important**
>
> En cas d'accès intercompte, un principal a besoin d'une `Allow` dans la politique d'identité intégrée et d'une politique basée sur les ressources.

Supposons qu'une politique basée sur une ressource accorde à tous les principaux de votre compte un accès administratif complet à une ressource. Vous pouvez ensuite déléguer l'accès complet, l'accès en lecture seule ou tout autre accès partiel aux principaux de votre compte. AWS Sinon, si la politique basée sur la ressource autorise uniquement les autorisations d'affichage de liste, vous pouvez déléguer uniquement ce type d'accès. Si vous essayez de déléguer davantage d'autorisations que n'en possède votre compte, l'accès de vos principaux restera limité à l'affichage de liste.

Pour plus d'informations sur la façon dont ces décisions sont prises, consultez Identification d'une demande autorisée ou refusée dans un compte.

> **Note**
>
> Les politiques IAM basées sur les ressources et les rôles ne délèguent l'accès entre les comptes qu'au sein d'une seule partition. Par exemple, vous ne pouvez pas ajouter d'accès entre comptes entre un compte dans la partition `aws` standard et un compte dans la partition `aws-cn`.

Par exemple, supposons que vous gériez `AccountA` et `AccountB`. Dans AccountA, vous avez un compartiment Amazon S3 nommé `BucketA`.



1. Vous attachez une politique basée sur les ressources à `BucketA` qui permet à tous les principaux de AccountB d'avoir un accès complet aux objets de votre compartiment. Ils peuvent créer, lire ou supprimer les objets de ce compartiment.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalAccess",
            "Effect": "Allow",
            "Principal": {"AWS": "arn:aws:iam::AccountB:root"},
```

```
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::BucketA/*"
            }
        ]
}
```

AccountA accorde à AccountB un accès complet à BucketA en désignant AccountB comme principal dans la politique basée sur les ressources. Par conséquent, AccountB est autorisé à effectuer toute action sur BucketA, et l'administrateur de AccountB peut déléguer l'accès à ses utilisateurs dans AccountB.

L'utilisateur root de AccountB dispose de toutes les autorisations accordées au compte. Par conséquent, l'utilisateur root a un accès complet à BucketA.

2. Dans AccountB, attachez une politique à l'utilisateur IAM nommé User2. Cette politique permet à l'utilisateur d'accéder en lecture seule aux objets de BucketA. Cela signifie que User2 peut afficher les objets, mais ne peut pas les créer, les modifier ou les supprimer.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect" : "Allow",
            "Action" : [
                "s3:Get*",
                "s3:List*" ],
                "Resource" : "arn:aws:s3:::BucketA/*"
        }
    ]
}
```

Le niveau d'accès maximal que AccountB peut déléguer est le niveau d'accès accordé au compte. Dans ce cas, la politique basée sur les ressources a accordé un accès complet à AccountB, mais User2 ne dispose que d'un accès en lecture seule.

L'administrateur de AccountB n'accorde pas l'accès à User1. Par défaut, les utilisateurs n'ont pas d'autorisations, sauf celles qui sont accordées explicitement, et ainsi User1 n'a pas accès à BucketA.

IAM évalue les autorisations d'un principal au moment où il fait une demande. Si vous utilisez des caractères génériques (*) pour donner aux utilisateurs un accès complet à vos ressources, les principaux peuvent accéder à toutes les ressources auxquelles votre AWS compte a accès. Cela est vrai même pour les ressources que vous ajoutez ou auxquelles vous accédez après la création de la politique de l'utilisateur.

Dans l'exemple précédent, si AccountB avait attaché à User2 une politique permettant un accès complet à toutes les ressources de tous les comptes, User2 aurait automatiquement eu accès à toutes les ressources auxquelles AccountB a accès. Cela inclut l'accès à BucketA et l'accès à toutes les autres ressources accordé par les politiques basées sur les ressources dans AccountA.

Pour plus d'informations sur l'utilisation des rôles de manière plus complexe, comme accorder l'accès à des applications et des services, consultez Scénarios courants pour les rôles : utilisateurs, applications et services.

> **Important**
>
> N'accordez l'accès qu'aux entités auxquelles vous faites confiance et accordez l'accès minimal nécessaire. Chaque fois que l'entité de confiance est un autre AWS compte, n'importe quel principal IAM peut avoir accès à votre ressource. Le AWS compte sécurisé ne peut déléguer l'accès que dans la mesure où l'accès lui a été accordé ; il ne peut pas déléguer un accès supérieur à celui accordé au compte lui-même.

Pour plus d'informations sur les autorisations, les politiques et le langage de politique d'autorisation que vous utilisez pour écrire des politiques, consultez Gestion de l'accès aux AWS ressources.

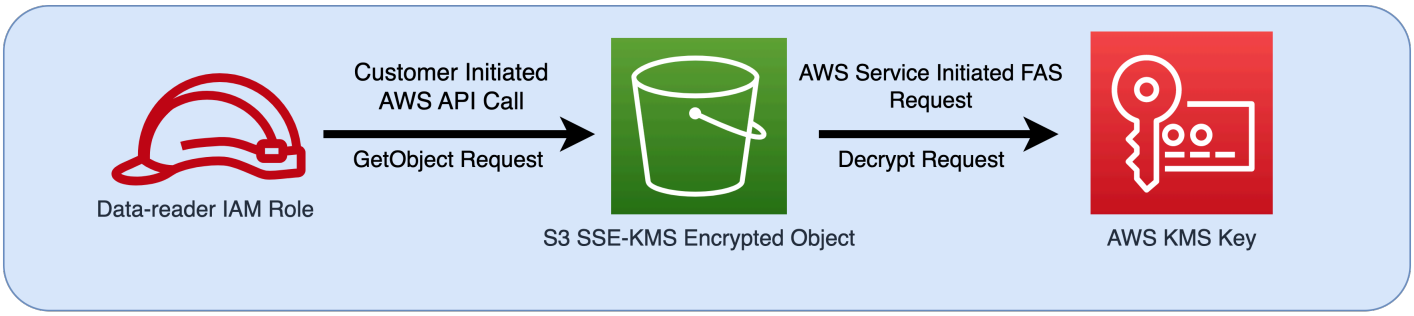
Transmission des sessions d'accès

Les sessions d'accès direct (FAS) sont une technologie IAM utilisée par les AWS services pour transmettre votre identité, vos autorisations et vos attributs de session lorsqu'un AWS service fait une demande en votre nom. FAS utilise les autorisations de l'identité appelant un AWS service, combinées à l'identité du AWS service, pour adresser des demandes aux services en aval. Les demandes FAS ne sont adressées aux AWS services pour le compte d'un principal IAM qu'une fois qu'un service a reçu une demande qui nécessite des interactions avec d'autres AWS services ou ressources pour être traitée. Lorsqu'une demande de FAS est effectuée :

- Le service qui reçoit la demande initiale d'un principal IAM vérifie les autorisations de ce dernier.

- Le service qui reçoit une demande de FAS ultérieure vérifie également les autorisations du même principal IAM.

Par exemple, le FAS est utilisé par Amazon S3 pour effectuer des appels AWS Key Management Service afin de déchiffrer un objet lorsque SSE-KMS a été utilisé pour le chiffrer. Lors du téléchargement d'un objet chiffré SSE-KMS, un rôle nommé lecteur de données appelle GetObject l'objet à Amazon S3, mais ne l'appelle pas directement. AWS KMS Après avoir reçu la GetObject demande et autorisé le lecteur de données, Amazon S3 envoie une demande FAS AWS KMS afin de déchiffrer l'objet Amazon S3. Lorsque KMS reçoit la demande de FAS, il vérifie les autorisations du rôle et n'autorise la demande de déchiffrement que si le lecteur de données dispose des autorisations correctes sur la clé KMS. Les demandes adressées à Amazon S3 et à Amazon S3 AWS KMS sont autorisées à l'aide des autorisations du rôle et ne sont couronnées de succès que si le lecteur de données est autorisé à la fois à accéder à l'objet Amazon S3 et à la clé AWS KMS.

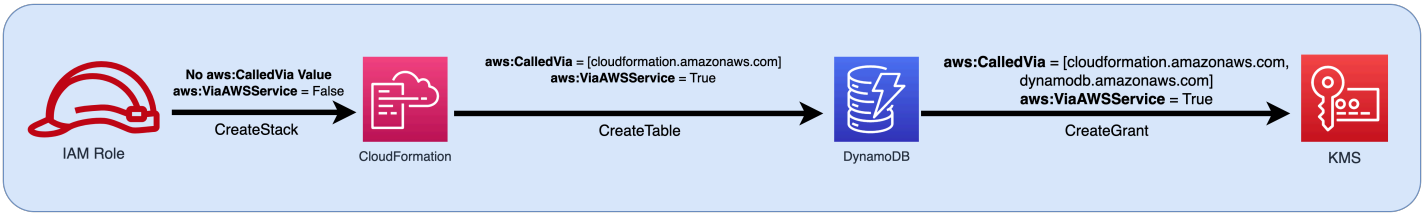


> **Note**
>
> Des demandes de FAS supplémentaires peuvent être effectuées par les services qui ont reçu une demande de FAS. Dans ce cas, le principal demandeur doit disposer d'autorisations pour tous les services appelés par la FAS.

Conditions des demandes de FAS et de la politique IAM

Lorsque des demandes de FAS sont effectuées, les clés de condition lois : CalledVia, aws : CalledVia Tout d'abord, et aws : CalledVia Dernier sont renseignées avec le principal du service qui a initié l'appel FAS. La valeur de la clé de condition AWS : via AWSService est définie sur `true` chaque fois qu'une demande de FAS est effectuée. Dans le schéma suivant, aucune `aws:CalledVia` clé de `aws:ViaAWSService` condition n'est définie pour la demande CloudFormation directe. Lorsque

CloudFormation DynamoDB envoie des demandes FAS en aval pour le compte du rôle, les valeurs de ces clés de condition sont renseignées.



Pour permettre à une demande de FAS d'être effectuée alors qu'elle serait autrement refusée par une déclaration de politique de refus avec une clé de condition testant les adresses IP source ou les VPC source, vous devez utiliser des clés de condition pour fournir une exception pour les demandes de FAS dans votre politique de refus. Cela peut être fait pour toutes les demandes de FAS en utilisant la clé de condition `aws:ViaAWSService`. Pour autoriser uniquement des AWS services spécifiques à effectuer des demandes FAS, utilisez`aws:CalledVia`.

> **Important**
>
> Lorsqu'une demande de FAS est effectuée après qu'une demande initiale a été effectuée via un point de terminaison d'un VPC, les valeurs des clés de condition pour `lois : SourceVpce`, `lois : SourceVpc` et `lois : VpcSource IP` de la demande initiale ne sont pas utilisées dans les demandes de FAS. Lorsque vous rédigez des politiques utilisant `aws:VPCSourceIP` ou `aws:SourceVPCE` pour accorder un accès conditionnel, vous devez également utiliser `aws:ViaAWSService` ou `aws:CalledVia` pour autoriser les demandes de FAS. Lorsqu'une demande FAS est faite après réception d'une demande initiale par un point de terminaison de AWS service public, les demandes FAS suivantes seront effectuées avec la même valeur de clé de `aws:SourceIP` condition.

Exemple : autoriser l'accès à Amazon S3 depuis un VPC ou avec la FAS

Dans l'exemple de politique IAM suivant, les demandes Amazon S3 GetObject et Athena ne sont autorisées que si elles proviennent de points de terminaison VPC attachés *à* example_vpc, ou s'il s'agit d'une demande FAS effectuée par Athena.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Sid": "OnlyAllowMyIPs",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*",
        "athena:StartQueryExecution",
        "athena:GetQueryResults",
        "athena:GetWorkGroup",
        "athena:StopQueryExecution",
        "athena:GetQueryExecution"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceVPC": [
            "example_vpc"
          ]
        }
      }
    },
    {
      "Sid": "OnlyAllowFAS",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": "athena.amazonaws.com"
        }
      }
    }
  ]
}
```

Pour d'autres exemples d'utilisation de clés de condition pour autoriser l'accès à la FAS, consultez le référentiel data perimeter example policy repo.

Exemples de politiques basées sur l'identité IAM

Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit leurs autorisations. AWS évalue ces politiques lorsqu'un principal IAM (utilisateur ou rôle) fait une demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée.

La plupart des politiques sont stockées AWS sous forme de documents JSON attachés à une identité IAM (utilisateur, groupe d'utilisateurs ou rôle). Les politiques basées sur l'identité incluent des politiques gérées par AWS des politiques gérées par le client et des politiques en ligne. Pour apprendre à créer une politique IAM à l'aide de ces exemples de document de politique JSON, consultez the section called “Création de politiques à l'aide de l'éditeur JSON”.

Par défaut, toutes les demandes sont refusées. Ainsi, vous devez fournir l'accès aux services, actions et ressources que vous avez l'intention d'autoriser en accès à l'identité. Si vous souhaitez également autoriser l'accès pour effectuer les actions spécifiées dans la console IAM, vous devez fournir des autorisations supplémentaires.

La bibliothèque de politiques suivante peut vous aider à définir des autorisations pour vos identités IAM. Une fois la politique recherchée trouvée, choisissez Afficher cette politique pour afficher le code JSON de la politique. Vous pouvez utiliser le document de politique JSON sous forme de modèle pour vos propres politiques.

> **Note**
>
> Si vous souhaitez envoyer une politique à inclure dans ce guide de référence, utilisez le bouton Commentaire situé au bas de cette page.

Exemples de politiques : AWS

- Autorise l'accès dans une plage de dates spécifique. (Afficher cette politique.)
- Permet d'activer et de désactiver AWS les régions. (Afficher cette politique.)
- Permet aux utilisateurs authentifiés MFA de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité. (Afficher cette politique.)
- Autorise un accès spécifique en cas d'utilisation de MFA pendant une plage de dates spécifique. (Afficher cette politique.)
- Permet aux utilisateurs de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité. (Afficher cette politique.)
- Permet aux utilisateurs de gérer leur propre appareil MFA sur la page Informations d'identification de sécurité. (Afficher cette politique.)
- Permet aux utilisateurs de gérer leur propre mot de passe sur la page Informations d'identification de sécurité. (Afficher cette politique.)

- Permet aux utilisateurs de gérer leur propre mot de passe, leurs clés d'accès et leurs clés publiques SSH sur la page Informations d'identification de sécurité. (Afficher cette politique.)
- Refuse l'accès à AWS en fonction de la région demandée. (Afficher cette politique.)
- Refuse l'accès à AWS en fonction de l'adresse IP source. (Afficher cette politique.)

Exemple de politique : AWS Data Exchange

- Refuser l'accès aux ressources Amazon S3 en dehors de votre compte, sauf AWS Data Exchange. (Afficher cette politique.)

Exemples de politiques : AWS Data Pipeline

- Refuse l'accès aux pipelines qu'un utilisateur n'a pas créés (Afficher cette politique).

Exemple de politiques : Amazon DynamoDB

- Autorise l'accès à une table Amazon DynamoDB spécifique (afficher cette politique).
- Autorise l'accès à des attributs Amazon DynamoDB spécifiques (afficher cette politique).
- Autorise l'accès au niveau des éléments à Amazon DynamoDB en fonction d'un ID Amazon Cognito (Afficher cette politique).

Exemples de politiques : Amazon EC2

- Autorise l'attachement ou le détachement de volumes Amazon EBS à des instances Amazon EC2 en fonction des balises (Afficher cette politique).
- Autorise le lancement d'instance Amazon EC2 dans un sous-réseau spécifique, par programmation et dans la console (Afficher cette politique)
- Autorise la gestion des groupes de sécurité Amazon EC2 attachés à un VPC spécifique, par programmation et dans la console (Afficher cette politique).
- Autorise le démarrage ou l'arrêt d'instances Amazon EC2 balisées par un utilisateur, par programmation et dans la console (Afficher cette politique).
- Autorise le démarrage ou l'arrêt d'instances Amazon EC2 en fonction des balises de ressource et de principal, par programmation et dans la console (Afficher cette politique).

- Autorise le démarrage ou l'arrêt des instances Amazon EC2 lorsque les balises de ressource et de principal correspondent (Afficher cette politique).
- Autorise l'accès Amazon EC2 complet dans une région spécifique, par programmation et dans la console. (Afficher cette politique.)
- Autorise le démarrage ou l'arrêt d'une instance Amazon EC2 et la modification d'un groupe de sécurité, par programmation et dans la console (Afficher cette politique).
- Refuse l'accès à des opérations Amazon EC2 spécifiques sans authentification MFA (afficher cette politique).
- Limite la suspension des instances Amazon EC2 à une plage d'adresses IP spécifiques (Afficher cette politique).

Exemples de politiques : AWS Identity and Access Management (IAM)

- Autorise l'accès à l'API du simulateur de politique (Afficher cette politique).
- Autorise l'accès à la console du simulateur de politique (Afficher cette politique).
- Permet d'endosser les rôles ayant une balise spécifique, par programmation et dans la console (Afficher cette politique).
- Autorise et refuse l'accès à plusieurs services, par programmation et dans la console (Afficher cette politique).
- Permet d'ajouter une balise spécifique à un utilisateur IAM avec une autre balise spécifique, par programmation et dans la console (afficher cette politique).
- Permet d'ajouter une balise spécifique à un utilisateur ou rôle IAM, par programmation et dans la console (afficher cette politique).
- Permet de créer un nouvel utilisateur uniquement avec des balises spécifiques (Afficher cette politique).
- Autorise la génération et l'extraction des rapports d'informations d'identification IAM (afficher cette politique).
- Autorise la gestion des membres d'un groupe, par programmation et dans la console (Afficher cette politique).
- Autorise la gestion d'une balise spécifique (Afficher cette politique).
- Autorise la transmission d'un rôle IAM à un service spécifique (afficher cette politique).
- Autorise l'accès en lecture seule à la console IAM sans notification (afficher cette politique).
- Autorise l'accès en lecture seule à la console IAM (afficher cette politique).

- Autorise des utilisateurs spécifiques à gérer un groupe, par programmation et dans la console (Afficher cette politique).
- Permet de définir les exigences de mot de passe de compte, par programmation et dans la console (Afficher cette politique).
- Autorise l'utilisation de l'API du simulateur de politiques pour les utilisateurs avec un chemin spécifique (Afficher cette politique).
- Autorise l'utilisation de la console du simulateur de politiques pour les utilisateurs avec un chemin spécifique (Afficher cette politique).
- Autorise les utilisateurs IAM à gérer eux-mêmes un dispositif MFA. (Afficher cette politique.)
- Autorise les utilisateurs IAM à définir leurs informations d'identification, par programmation et dans la console. (Afficher cette politique.)
- Permet d'afficher les informations du dernier accès au service pour une AWS Organizations politique dans la console IAM. (Afficher cette politique.)
- Limite les politiques gérées pouvant être appliquées à un nouvel utilisateur, groupe ou rôle IAM (afficher cette politique).
- Autorise l'accès aux politiques IAM uniquement dans votre compte (Afficher cette politique.)

Exemples de politiques : AWS Lambda

- Permet à une AWS Lambda fonction d'accéder à une table Amazon DynamoDB (consultez cette politique.)

Exemple de politiques : Amazon RDS

- Autorise l'accès complet à la base de données Amazon RDS dans une région donnée. (Afficher cette politique.)
- Autorise la restauration des bases de données Amazon RDS, par programmation et dans la console (afficher cette politique).
- Accorde aux propriétaires de balise l'accès complet aux ressources Amazon RDS qu'ils ont balisées (afficher cette politique).

Exemples de politiques : Amazon S3

- Autorise un utilisateur Amazon Cognito à accéder aux objets dans leur propre compartiment Amazon S3 (Afficher cette politique).
- Autorise les utilisateurs fédérés à accéder à leur répertoire de base dans Amazon S3, par programmation et dans la console (afficher cette politique).
- Autorise l'accès S3 complet, mais refuse explicitement l'accès au compartiment de production si l'administrateur ne s'est pas connecté à l'aide d'une autorisation MFA au cours des 30 dernières minutes (Afficher cette politique).
- Autorise les utilisateurs IAM à accéder à leurs répertoire d'accueil dans Amazon S3, par programmation et dans la console (Afficher cette politique)
- Permet à un utilisateur de gérer un seul compartiment Amazon S3 et refuse toutes les autres AWS actions et ressources (consultez cette politique.)
- Autorise l'accès en `Read` et en `Write` à un compartiment Amazon S3 spécifique (Afficher cette politique).
- Autorise l'accès en `Read` et en `Write` à un compartiment Amazon S3 spécifique, par programmation et dans la console (Afficher cette politique).

AWS : permet l'accès en fonction de la date et de l'heure

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès aux actions en fonction de la date et de l'heure. Cette politique limite l'accès aux actions effectuées entre le 1er avril 2020 et le 30 juin 2020 (UTC) inclus. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Pour en savoir plus sur l'utilisation de plusieurs conditions au sein du bloc `Condition` d'une politique IAM, consultez Plusieurs valeurs dans un élément Condition.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "service-prefix:action-name",
```

```
            "Resource": "*",
            "Condition": {
                "DateGreaterThan": {"aws:CurrentTime": "2020-04-01T00:00:00Z"},
                "DateLessThan": {"aws:CurrentTime": "2020-06-30T23:59:59Z"}
            }
        }
    ]
}
```

> **Note**
>
> Vous ne pouvez pas utiliser une variable de politique avec l'opérateur de condition Date. Pour en savoir plus, consultez Élément de condition

AWS: permet d'activer et de désactiver AWS les régions

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un administrateur à activer et désactiver la région Asie-Pacifique (Hong Kong) (ap-east-1). Cette politique définit des autorisations pour l'accès à la console et par programmation. Ce paramètre s'affiche dans la page Paramètres du compte dans la AWS Management Console. Cette page comprend les informations de niveau compte sensibles qui doivent être affichées et gérées uniquement par les administrateurs de compte. Pour utiliser cette politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

> **Important**
>
> Vous ne pouvez pas activer ou désactiver les régions qui sont activées par défaut. Vous pouvez uniquement inclure les régions désactivées par défaut. Pour plus d'informations, consultez Gestion des régions AWS dans le Références générales AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EnableDisableHongKong",
```

```
            "Effect": "Allow",
            "Action": [
                "account:EnableRegion",
                "account:DisableRegion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {"account:TargetRegion": "ap-east-1"}
            }
        },
        {
            "Sid": "ViewConsole",
            "Effect": "Allow",
            "Action": [
                "account:ListRegions"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS: permet aux utilisateurs IAM authentifiés par MFA de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité

Cet exemple montre comment créer une politique basée sur l'identité qui permet aux utilisateurs IAM authentifiés à l'aide de l'authentification multifactorielle (MFA) de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité. Cette page AWS Management Console affiche les informations relatives au compte, telles que l'ID de compte et l'ID d'utilisateur canonique. Les utilisateurs peuvent également consulter et modifier leurs mots de passe, leurs clés d'accès, leurs dispositifs MFA, leurs certificats X.509, leurs clés SSH et leurs informations d'identification Git. Cet exemple de politique inclut les autorisations requises pour consulter et modifier toutes les informations de la page. Cela oblige également l'utilisateur à configurer et à s'authentifier à l'aide de la MFA avant d'effectuer toute autre opération dans. AWS Pour autoriser les utilisateurs à gérer leurs propres informations d'identification sans utiliser l'authentification MFA, consultez AWS: permet aux utilisateurs IAM de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

Pour savoir comment les utilisateurs peuvent accéder à la page des informations d'identification de sécurité, voirComment les utilisateurs IAM modifient leur mot de passe (console).

> **Note**
>
> - Cet exemple de politique n'autorise pas les utilisateurs à réinitialiser un mot de passe lorsqu'ils se connectent AWS Management Console pour la première fois. Nous vous recommandons de n'accorder des autorisations aux nouveaux utilisateurs qu'après leur enregistrement. Pour plus d'informations, consultez Comment créer des utilisateurs IAM en toute sécurité ?. Cela empêche également les utilisateurs dont le mot de passe a expiré de le réinitialiser pendant la procédure d'enregistrement. Vous pouvez autoriser cette opération en ajoutant `iam:ChangePassword` et `iam:GetAccountPasswordPolicy` à l'instruction `DenyAllExceptListedIfNoMFA`. Cependant, nous ne le recommandons pas, car autoriser les utilisateurs à modifier leur mot de passe sans MFA peut poser des problèmes de sécurité.
> - Si vous avez l'intention d'utiliser cette stratégie pour l'accès par programmation, vous devez appeler `GetSessionToken` pour une authetification auprès de MFA. Pour plus d'informations, consultez Configuration de l'accès aux API protégé par MFA.

À quoi sert cette politique ?

- L'instruction `AllowViewAccountInfo` permet à l'utilisateur d'afficher les informations au niveau du compte. Ces autorisations doivent se trouver dans leur propre instruction, car elles ne prennent pas en charge ou n'ont pas besoin de spécifier d'ARN de ressource particulier. À la place, les autorisations spécifient `"Resource" : "*"`. Cette instruction contient les actions suivantes qui permettent à l'utilisateur d'afficher des informations spécifiques :
  - `GetAccountPasswordPolicy` : afficher les exigences de mot de passe du compte lors de la modification de son propre mot de passe utilisateur IAM.
  - `ListVirtualMFADevices` : afficher les détails relatifs à un dispositif MFA virtuel qui est activé pour l'utilisateur.
- L'instruction `AllowManageOwnPasswords` permet à l'utilisateur de modifier son propre mot de passe. Cette instruction inclut également l'action `GetUser`, qui est requise pour afficher la plupart des informations de la page My security credentials (Mes informations d'identification de sécurité).
- L'instruction `AllowManageOwnAccessKeys` permet à l'utilisateur de créer, mettre à jour et supprimer ses propres clés d'accès. L'utilisateur récupère peut également récupérer des informations relatives au moment où la clé d'accès spécifiée a été utilisée pour la dernière fois.

- L'instruction `AllowManageOwnSigningCertificates` permet à l'utilisateur de charger, mettre à jour et supprimer ses propres certificats de signature.
- L'`AllowManageOwnSSHPublicKeys`instruction permet à l'utilisateur de télécharger, de mettre à jour et de supprimer ses propres clés publiques SSH pour CodeCommit.
- L'`AllowManageOwnGitCredentials`instruction permet à l'utilisateur de créer, de mettre à jour et de supprimer ses propres informations d'identification Git pour CodeCommit.
- L'instruction `AllowManageOwnVirtualMFADevice` permet à l'utilisateur de créer son propre dispositif MFA. L'ARN de ressource dans l'instruction autorise uniquement l'utilisateur à créer un dispositif MFA de n'importe quel nom, mais les autres instructions de la politique autorisent uniquement l'utilisateur à connecter le dispositif à l'utilisateur actuellement connecté.
- L'instruction `AllowManageOwnUserMFA` permet à l'utilisateur de consulter et gérer le dispositif MFA matériel, U2F ou virtuel pour leur propre utilisateur. L'ARN de ressource dans l'instruction autorise uniquement l'accès au propre utilisateur IAM de l'utilisateur. Les utilisateurs ne peuvent pas afficher ni gérer le dispositif MFA pour d'autres utilisateurs.
- La `DenyAllExceptListedIfNoMFA` déclaration refuse l'accès à toutes les actions dans tous les AWS services, à l'exception de quelques actions répertoriées, mais uniquement si l'utilisateur n'est pas connecté avec le MFA. L'instruction utilise une combinaison de `"Deny"` et `"NotAction"` pour refuser explicitement l'accès à toutes les actions non répertoriées. Les éléments répertoriés ne sont pas refusés ou autorisés par cette instruction. Toutefois, les actions sont autorisées par d'autres instructions de la politique. Pour plus d'informations sur la logique de cette instruction, consultez la section NotAction avec Deny. Si l'utilisateur est connecté avec MFA, le test `Condition` échoue et cette instruction ne refuse aucune action. Dans ce cas, les autorisations de l'utilisateur sont définies par d'autres stratégies ou instructions.

  Cette instruction garantit à l'utilisateur de pouvoir effectuer uniquement les actions répertoriées lorsqu'il n'est pas connecté avec MFA. Par ailleurs, il peut effectuer les actions répertoriées uniquement si une autre instruction ou politique permet l'accès à ces actions. Cela ne permet pas à un utilisateur de créer un mot de passe lors de la connexion, car l'action `iam:ChangePassword` ne doit pas être autorisée sans l'authentification MFA.

  La version `...IfExists` de l'opérateur `Bool` permet de s'assurer que si la clé `lois : MultiFactor AuthPresent` est manquante, la condition renvoie la valeur true. Cela signifie qu'un utilisateur qui accède à une API avec des informations d'identification à long terme, comme une clé d'accès, se voit refuser l'accès aux opérations d'API non IAM.

Cette politique ne permet pas aux utilisateurs d'afficher la page Users (Utilisateurs) dans la console IAM, ou d'utiliser cette page pour accéder à leurs propres informations utilisateur. Pour que cela soit possible, ajoutez l'action `iam:ListUsers` à l'instruction `AllowViewAccountInfo` et à l'instruction `DenyAllExceptListedIfNoMFA`. Cette stratégie ne permet pas non plus aux utilisateurs de modifier leur mot de passe sur leur propre page utilisateur. Pour que cela soit possible, ajoutez les actions `iam:GetLoginProfile` et `iam:UpdateLoginProfile` à l'instruction `AllowManageOwnPasswords`. Par ailleurs, pour autoriser un utilisateur à modifier son mot de passe dans sa propre page utilisateur sans se connecter à l'aide de MFA, ajoutez l'action `iam:UpdateLoginProfile` à l'instruction `DenyAllExceptListedIfNoMFA`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
```

```
        },
        {
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
                "iam:UploadSigningCertificate"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnGitCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:ResetServiceSpecificCredential",
                "iam:UpdateServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
```

```
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

AWS : autorise l'accès spécifique dans un délai spécifique à l'aide de MFA

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui utilise plusieurs conditions qui sont évaluées à l'aide d'un opérateur logique `AND`. Il permet l'accès complet au service nommé `SERVICE-NAME-1`, et l'accès aux actions `ACTION-NAME-A` et `ACTION-NAME-B` dans le service nommé `SERVICE-NAME-2`. Ces actions sont autorisées uniquement lorsque l'utilisateur est authentifié à l'aide de l'authentification multifacteur (Multi-Factor Authentication, MFA). L'accès est limité aux actions se produisant entre le 1er juillet 2017 et le 31 décembre 2017

(UTC) inclus. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Pour en savoir plus sur l'utilisation de plusieurs conditions au sein du bloc `Condition` d'une politique IAM, consultez Plusieurs valeurs dans un élément Condition.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "service-prefix-1:*",
            "service-prefix-2:action-name-a",
            "service-prefix-2:action-name-b"
        ],
        "Resource": "*",
        "Condition": {
            "Bool": {"aws:MultiFactorAuthPresent": true},
            "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
            "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
        }
    }
}
```

AWS: permet aux utilisateurs IAM de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité

Cet exemple montre comment créer une politique basée sur l'identité qui permet aux utilisateurs IAM de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité. Cette AWS Management Console page affiche les informations du compte, telles que l'identifiant du compte et l'identifiant utilisateur canonique. Les utilisateurs peuvent également consulter et modifier leurs propres mots de passe, clés d'accès, certificats X.509, clés SSH et informations d'identification Git. Cet exemple de politique inclut les autorisations requises pour consulter et modifier toutes les informations de la page sauf le dispositif MFA de l'utilisateur. Pour autoriser les utilisateurs à gérer leurs propres informations d'identification avec l'authentification MFA, consultez AWS: permet aux utilisateurs IAM authentifiés par MFA de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

Pour savoir comment les utilisateurs peuvent accéder à la page des informations d'identification de sécurité, voirComment les utilisateurs IAM modifient leur mot de passe (console).

À quoi sert cette politique ?

- L'instruction `AllowViewAccountInfo` permet à l'utilisateur d'afficher les informations au niveau du compte. Ces autorisations doivent se trouver dans leur propre instruction, car elles ne prennent pas en charge ou n'ont pas besoin de spécifier d'ARN de ressource particulier. À la place, les autorisations spécifient `"Resource" : "*"`. Cette instruction contient les actions suivantes qui permettent à l'utilisateur d'afficher des informations spécifiques :
  - `GetAccountPasswordPolicy` : afficher les exigences de mot de passe du compte lors de la modification de son propre mot de passe utilisateur IAM.
  - `GetAccountSummary` : afficher l'ID du compte et l'ID d'utilisateur canonique du compte.
- L'instruction `AllowManageOwnPasswords` permet à l'utilisateur de modifier son propre mot de passe. Cette instruction inclut également l'action `GetUser`, qui est requise pour afficher la plupart des informations de la page My security credentials (Mes informations d'identification de sécurité).
- L'instruction `AllowManageOwnAccessKeys` permet à l'utilisateur de créer, mettre à jour et supprimer ses propres clés d'accès. L'utilisateur récupère peut également récupérer des informations relatives au moment où la clé d'accès spécifiée a été utilisée pour la dernière fois.
- L'instruction `AllowManageOwnSigningCertificates` permet à l'utilisateur de charger, mettre à jour et supprimer ses propres certificats de signature.
- L'`AllowManageOwnSSHPublicKeys`instruction permet à l'utilisateur de télécharger, de mettre à jour et de supprimer ses propres clés publiques SSH pour CodeCommit.
- L'`AllowManageOwnGitCredentials`instruction permet à l'utilisateur de créer, de mettre à jour et de supprimer ses propres informations d'identification Git pour CodeCommit.

Cette politique ne permet pas aux utilisateurs de consulter ni de gérer leurs propres dispositifs MFA. Ils ne peuvent pas afficher la page Users (Utilisateurs) dans la console IAM, ou utiliser cette page pour accéder à leurs propres informations utilisateur. Pour que cela soit possible, ajoutez l'action `iam:ListUsers` à l'instruction `AllowViewAccountInfo`. Cette stratégie ne permet pas non plus aux utilisateurs de modifier leur mot de passe sur leur propre page utilisateur. Pour que cela soit possible, ajoutez les actions `iam:CreateLoginProfile`, `iam:DeleteLoginProfile`, `iam:GetLoginProfile` et `iam:UpdateLoginProfile` à l'instruction `AllowManageOwnPasswords`.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
                "iam:UploadSigningCertificate"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
```

```
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnGitCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:ResetServiceSpecificCredential",
                "iam:UpdateServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: permet aux utilisateurs IAM authentifiés MFA de gérer leur propre appareil MFA sur la page des informations d'identification de sécurité

Cet exemple montre comment créer une politique basée sur l'identité qui permet aux utilisateurs IAM authentifiés par le biais de l'authentification multifactorielle (MFA) de gérer leur propre appareil MFA sur la page Informations d'identification de sécurité. Cette AWS Management Console page affiche les informations relatives au compte et à l'utilisateur, mais l'utilisateur ne peut consulter et modifier que son propre dispositif MFA. Pour autoriser les utilisateurs à gérer toutes leurs propres informations d'identification avec l'authentification MFA, consultez AWS: permet aux utilisateurs IAM authentifiés par MFA de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

> **Note**
>
> Si un utilisateur IAM doté de cette politique n'est pas authentifié par MFA, cette politique refuse l'accès à toutes les AWS actions, à l'exception de celles nécessaires pour s'authentifier à l'aide de l'authentification MFA. Pour utiliser l' AWS API AWS CLI et, les utilisateurs IAM doivent d'abord récupérer leur jeton MFA à l'aide de AWS STS GetSessionTokenl'opération, puis utiliser ce jeton pour authentifier l'opération souhaitée. D'autres politiques, telles que les politiques basées sur les ressources ou d'autres politiques basées sur l'identité peuvent autoriser des actions dans d'autres services. Cette politique refusera cet accès si l'utilisateur IAM n'est pas authentifié avec MFA.

Pour savoir comment les utilisateurs peuvent accéder à la page des informations d'identification de sécurité, voirComment les utilisateurs IAM modifient leur mot de passe (console).

À quoi sert cette politique ?

- L'instruction `AllowViewAccountInfo` permet à l'utilisateur d'afficher les détails relatifs à un dispositif MFA virtuel, qui est activé pour l'utilisateur. Cette autorisation doit se trouver dans sa propre instruction, car elle ne prend pas en charge la spécification d'un ARN de ressource. À la place, vous devez spécifier `"Resource" : "*"`.
- L'instruction `AllowManageOwnVirtualMFADevice` permet à l'utilisateur de créer son propre dispositif MFA. L'ARN de ressource dans l'instruction autorise uniquement l'utilisateur à créer un dispositif MFA de n'importe quel nom, mais les autres instructions de la politique autorisent uniquement l'utilisateur à connecter le dispositif à l'utilisateur actuellement connecté.
- L'instruction `AllowManageOwnUserMFA` permet à l'utilisateur de consulter et gérer son propre dispositif MFA matériel, U2F ou virtuel. L'ARN de ressource dans l'instruction autorise uniquement l'accès au propre utilisateur IAM de l'utilisateur. Les utilisateurs ne peuvent pas afficher ni gérer le dispositif MFA pour d'autres utilisateurs.
- La `DenyAllExceptListedIfNoMFA` déclaration refuse l'accès à toutes les actions dans tous les AWS services, à l'exception de quelques actions répertoriées, mais uniquement si l'utilisateur n'est pas connecté avec le MFA. L'instruction utilise une combinaison de `"Deny"` et `"NotAction"` pour refuser explicitement l'accès à toutes les actions non répertoriées. Les éléments répertoriés ne sont pas refusés ou autorisés par cette instruction. Toutefois, les actions sont autorisées par d'autres instructions de la politique. Pour plus d'informations sur la logique de cette instruction, consultez la section NotAction avec Deny. Si l'utilisateur est connecté avec

MFA, le test `Condition` échoue et cette instruction ne refuse aucune action. Dans ce cas, les autorisations de l'utilisateur sont définies par d'autres stratégies ou instructions.

Cette instruction garantit à l'utilisateur de pouvoir effectuer uniquement les actions répertoriées lorsqu'il n'est pas connecté avec MFA. Par ailleurs, il peut effectuer les actions répertoriées uniquement si une autre instruction ou politique permet l'accès à ces actions.

La version `...IfExists` de l'opérateur `Bool` permet de s'assurer que si la clé `aws:MultiFactorAuthPresent` est manquante, la condition renvoie la valeur true. Cela signifie qu'un utilisateur qui accède à une opération d'API avec des informations d'identification à long terme, comme une clé d'accès, se voit refuser l'accès aux opérations d'API non IAM.

Cette politique ne permet pas aux utilisateurs d'afficher la page Users (Utilisateurs) dans la console IAM, ou d'utiliser cette page pour accéder à leurs propres informations utilisateur. Pour que cela soit possible, ajoutez l'action `iam:ListUsers` à l'instruction `AllowViewAccountInfo` et à l'instruction `DenyAllExceptListedIfNoMFA`.

> **Warning**
>
> N'ajoutez pas d'autorisations pour supprimer un dispositif MFA sans authentification MFA. Les utilisateurs dotés de cette politique peuvent tenter de s'attribuer un dispositif MFA virtuel et recevoir une erreur indiquant qu'ils ne sont pas autorisés à effectuer `iam:DeleteVirtualMFADevice`. Si cela se produit, n'ajoutez pas cette autorisation à l'instruction `DenyAllExceptListedIfNoMFA`. Les utilisateurs qui ne sont pas authentifiés avec MFA ne doivent jamais être autorisés à supprimer un dispositif MFA. Les utilisateurs peuvent consulter cette erreur s'ils ont précédemment commencé à affecter un dispositif MFA virtuel à leur utilisateur et annulé le processus. Pour résoudre ce problème, vous ou un autre administrateur devez supprimer le dispositif MFA virtuel existant de l'utilisateur à l'aide de l'API AWS CLI or AWS . Pour plus d'informations, voir Je ne suis pas autorisé à effectuer : iam : MFAdevice DeleteVirtual.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
```

```
            "Action": "iam:ListVirtualMFADevices",
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}
            }
        }
    ]
}
```

AWS: permet aux utilisateurs IAM de modifier leur propre mot de passe de console sur la page des informations d'identification de sécurité

Cet exemple montre comment créer une politique basée sur l'identité qui permet aux utilisateurs IAM de modifier leur propre AWS Management Console mot de passe sur la page Informations d'identification de sécurité. Cette AWS Management Console page affiche les informations relatives au compte et à l'utilisateur, mais l'utilisateur ne peut accéder qu'à son propre mot de passe. Pour autoriser les utilisateurs à gérer toutes leurs propres informations d'identification avec l'authentification MFA, consultez AWS: permet aux utilisateurs IAM authentifiés par MFA de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité. Pour autoriser les utilisateurs à gérer leurs propres informations d'identification sans utiliser l'authentification MFA, consultez AWS: permet aux utilisateurs IAM de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

Pour savoir comment les utilisateurs peuvent accéder à la page des informations d'identification de sécurité, voirComment les utilisateurs IAM modifient leur mot de passe (console).

À quoi sert cette politique ?

- L'instruction `ViewAccountPasswordRequirements` autorise l'utilisateur à afficher les exigences de mot de passe du compte lors de la modification de son propre mot de passe utilisateur IAM.
- L'instruction `ChangeOwnPassword` permet à l'utilisateur de modifier son propre mot de passe. Cette instruction inclut également l'action `GetUser`, qui est requise pour afficher la plupart des informations de la page My security credentials (Mes informations d'identification de sécurité).

Cette politique ne permet pas aux utilisateurs d'afficher la page Users (Utilisateurs) dans la console IAM, ou d'utiliser cette page pour accéder à leurs propres informations utilisateur. Pour que cela soit possible, ajoutez l'action `iam:ListUsers` à l'instruction `ViewAccountPasswordRequirements`. Cette stratégie ne permet pas non plus aux utilisateurs de modifier leur mot de passe sur leur propre page utilisateur. Pour que cela soit possible, ajoutez les actions `iam:GetLoginProfile` et `iam:UpdateLoginProfile` à l'instruction `ChangeOwnPasswords`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewAccountPasswordRequirements",
            "Effect": "Allow",
            "Action": "iam:GetAccountPasswordPolicy",
```

```
            "Resource": "*"
        },
        {
            "Sid": "ChangeOwnPassword",
            "Effect": "Allow",
            "Action": [
                "iam:GetUser",
                "iam:ChangePassword"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: permet aux utilisateurs IAM de gérer leur propre mot de passe, leurs clés d'accès et leurs clés publiques SSH sur la page Informations d'identification de sécurité

Cet exemple montre comment créer une politique basée sur l'identité qui permet aux utilisateurs IAM de gérer leur propre mot de passe, leurs clés d'accès et leurs certificats X.509 sur la page Informations d'identification de sécurité. Cette page AWS Management Console affiche les informations relatives au compte, telles que l'ID de compte et l'ID d'utilisateur canonique. Les utilisateurs peuvent également consulter et modifier leurs propres mots de passe, clés d'accès, dispositifs MFA, certificats X.509, clés SSH et informations d'identification Git. Cet exemple de politique inclut les autorisations requises pour consulter et modifier uniquement le mot de passe, les clés d'accès et le certificat X.509. Pour autoriser les utilisateurs à gérer toutes leurs propres informations d'identification avec l'authentification MFA, consultez AWS: permet aux utilisateurs IAM authentifiés par MFA de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité. Pour autoriser les utilisateurs à gérer leurs propres informations d'identification sans utiliser l'authentification MFA, consultez AWS: permet aux utilisateurs IAM de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

Pour savoir comment les utilisateurs peuvent accéder à la page des informations d'identification de sécurité, voirComment les utilisateurs IAM modifient leur mot de passe (console).

À quoi sert cette politique ?

- L'instruction `AllowViewAccountInfo` permet à l'utilisateur d'afficher les informations au niveau du compte. Ces autorisations doivent se trouver dans leur propre instruction, car elles ne prennent pas en charge ou n'ont pas besoin de spécifier d'ARN de ressource particulier. À la place, les

autorisations spécifient `"Resource" : "*"`. Cette instruction contient les actions suivantes qui permettent à l'utilisateur d'afficher des informations spécifiques :

  - `GetAccountPasswordPolicy` : afficher les exigences de mot de passe du compte lors de la modification de son propre mot de passe utilisateur IAM.
  - `GetAccountSummary` : afficher l'ID du compte et l'ID d'utilisateur canonique du compte.
- L'instruction `AllowManageOwnPasswords` permet à l'utilisateur de modifier son propre mot de passe. Cette instruction inclut également l'action `GetUser`, qui est requise pour afficher la plupart des informations de la page My security credentials (Mes informations d'identification de sécurité).
- L'instruction `AllowManageOwnAccessKeys` permet à l'utilisateur de créer, mettre à jour et supprimer ses propres clés d'accès. L'utilisateur récupère peut également récupérer des informations relatives au moment où la clé d'accès spécifiée a été utilisée pour la dernière fois.
- L'`AllowManageOwnSSHPublicKeys`instruction permet à l'utilisateur de télécharger, de mettre à jour et de supprimer ses propres clés publiques SSH pour CodeCommit.

Cette politique ne permet pas aux utilisateurs de consulter ni de gérer leurs propres dispositifs MFA. Ils ne peuvent pas afficher la page Users (Utilisateurs) dans la console IAM, ou utiliser cette page pour accéder à leurs propres informations utilisateur. Pour que cela soit possible, ajoutez l'action `iam:ListUsers` à l'instruction `AllowViewAccountInfo`. Cette stratégie ne permet pas non plus aux utilisateurs de modifier leur mot de passe sur leur propre page utilisateur. Pour que cela soit possible, ajoutez les actions `iam:GetLoginProfile` et `iam:UpdateLoginProfile` à l'instruction `AllowManageOwnPasswords`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
```

```
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: refuse l'accès AWS en fonction de la région demandée

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui refuse l'accès à toutes les actions en dehors des régions spécifiées avec la clé de condition `aws:RequestedRegion`, à l'exception des actions dans les services spécifiés avec `NotAction`. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Cette politique utilise l'élément `NotAction` avec l'effet `Deny`, qui refuse explicitement l'accès à toutes les actions non répertoriées dans l'instruction. Les actions relatives à l' CloudFrontIAM, à Route 53 et AWS Support aux services ne doivent pas être refusées, car il s'agit de services AWS mondiaux populaires dotés d'un point de terminaison unique situé physiquement dans la `us-east-1` région. Étant donné que toutes les demandes adressées à ces services sont effectuées vers la région `us-east-1`, les demandes sont refusées sans l'élément `NotAction`. Modifiez cet élément pour inclure les actions pour d'autres services AWS globaux que vous utilisez, tels que `budgets`, `globalaccelerator`, `importexport`, `organizations` ou `waf`. Certains autres services mondiaux, tels que AWS Chatbot et AWS Device Farm, sont des services mondiaux dont les points de terminaison sont physiquement situés dans la `us-west-2` région. Pour en savoir plus sur tous les services qui ont un seul point de terminaison global, consultez AWS Régions et points de terminaison dans le document Références générales AWS. Pour de plus amples informations sur l'utilisation de l'élément `NotAction` avec l'effet `Deny`, veuillez consulter Éléments de politique JSON IAM : NotAction.

> **Important**
>
> Cette politique ne permet aucune action. Utilisez cette stratégie conjointement à d'autres stratégies qui autorisent des actions spécifiques.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAllOutsideRequestedRegions",
            "Effect": "Deny",
            "NotAction": [
                "cloudfront:*",
                "iam:*",
                "route53:*",
                "support:*"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestedRegion": [
                        "eu-central-1",
                        "eu-west-1",
                        "eu-west-2",
```

```
                    "eu-west-3"
                ]
            }
        }
    }
]
}
```

AWS: refuse l'accès à la AWS base de l'adresse IP source

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui refuse l'accès à toutes les AWS actions du compte lorsque la demande provient de principaux situés en dehors de la plage d'adresses IP spécifiée. Cette politique est utile lorsque les adresses IP de votre entreprise se situent dans les plages d'adresses IP spécifiées. Dans cet exemple, la demande sera rejetée sauf si elle provient de la plage CIDR 192.0.2.0/24 ou 203.0.113.0/24. La politique ne refuse pas les demandes effectuées par les AWS services utilisateurs, Transmission des sessions d'accès car l'adresse IP du demandeur d'origine est préservée.

Soyez prudent lorsque vous utilisez des conditions négatives dans la même instruction de politique comme `"Effect": "Deny"`. Dans ce cas, les actions spécifiées dans l'instruction de politique sont explicitement refusées dans toutes les conditions sauf celles spécifiées.

> **Important**
>
> Cette politique ne permet aucune action. Utilisez cette stratégie conjointement à d'autres stratégies qui autorisent des actions spécifiques.

Lorsque d'autres politiques autorisent des actions, les principaux peuvent effectuer des demandes à partir de la plage d'adresses IP. Un AWS service peut également effectuer des demandes en utilisant les informations d'identification du principal. Lorsqu'un principal effectue une demande en dehors de la plage d'adresses IP, la demande est refusée.

Pour de plus amples informations sur l'utilisation de la clé de condition `aws:SourceIp`, en particulier sur le moment où `aws:SourceIp` risque de ne pas fonctionne dans votre politique, veuillez consulter AWS clés contextuelles de condition globale.

```
{
    "Version": "2012-10-17",
    "Statement": {
```

```
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
            "NotIpAddress": {
                "aws:SourceIp": [
                    "192.0.2.0/24",
                    "203.0.113.0/24"
                ]
            }
        }
    }
}
```

AWS: Refusez l'accès aux ressources Amazon S3 en dehors de votre compte, sauf AWS Data Exchange

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui refuse l'accès à toutes les ressources AWS qui n'appartiennent pas à votre compte, à l'exception des ressources nécessaires au fonctionnement normal. AWS Data Exchange Pour utiliser cette politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Vous pouvez créer une politique similaire pour restreindre l'accès aux ressources au sein d'une organisation ou d'une unité organisationnelle, tout en comptabilisant les ressources AWS Data Exchange détenues à l'aide des clés de condition `aws:ResourceOrgPaths` et`aws:ResourceOrgID`.

Si vous l'utilisez AWS Data Exchange dans votre environnement, le service crée et interagit avec des ressources telles que les compartiments Amazon S3 appartenant au compte de service. Par exemple, AWS Data Exchange envoie des demandes aux compartiments Amazon S3 appartenant au AWS Data Exchange service pour le compte du principal IAM (utilisateur ou rôle) invoquant les API. AWS Data Exchange Dans ce cas, l'utilisation `aws:ResourceAccount`aws:ResourceOrgPaths, ou `aws:ResourceOrgID` dans le cadre d'une politique, sans tenir compte des ressources AWS Data Exchange détenues, empêche l'accès aux compartiments appartenant au compte de service.

- Instruction `DenyAllAwsResourcesOutsideAccountExceptS3` utilise le l'élément `NotAction` avec l'effet Deny (Refuser) qui refuse explicitement l'accès à toutes les actions non répertoriées dans l'instruction et qui n'appartiennent pas non plus au compte répertorié. L'élément `NotAction`

indique les exceptions à cette instruction. Ces actions constituent une exception à cette déclaration, car si elles sont effectuées sur des ressources créées par AWS Data Exchange, la politique les refuse.

- L'instruction `DenyAllS3ResoucesOutsideAccountExceptDataExchange` utilise une combinaison des clés de condition `ResourceAccount` et `CalledVia` pour refuser l'accès aux trois actions Amazon S3 exclues dans la déclaration précédente. L'instruction refuse les actions si les ressources n'appartiennent pas au compte répertorié et si le service appelant n'est pas AWS Data Exchange. L'instruction ne refuse pas les actions si la ressource appartient au compte répertorié ou si le principal de service répertorié, `dataexchange.amazonaws.com`, effectue les opérations.

> **Important**
>
> Cette politique ne permet aucune action. Elle utilise l'effet `Deny`, qui refuse explicitement l'accès à toutes les ressources répertoriées dans l'instruction n'appartenant pas au compte répertorié. Utilisez cette politique en combinaison avec d'autres politiques qui autorisent l'accès à des ressources spécifiques.

L'exemple suivant montre comment vous pouvez configurer la politique pour autoriser l'accès aux compartiments Amazon S3 requis.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAllAwsReourcesOutsideAccountExceptAmazonS3",
      "Effect": "Deny",
      "NotAction": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
```

```
        }
      }
    },
    {
      "Sid": "DenyAllS3ResourcesOutsideAccountExceptDataExchange",
      "Effect": "Deny",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        },
        "ForAllValues:StringNotEquals": {
          "aws:CalledVia": [
            "dataexchange.amazonaws.com"
          ]
        }
      }
    }
  ]
}
```

AWS Data Pipeline: refuse l'accès aux DataPipeline pipelines qu'un utilisateur n'a pas créés

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui refuse l'accès aux pipelines qu'un utilisateur n'a pas créés. Si la valeur du champ `PipelineCreator` correspond au nom d'utilisateur IAM, alors les actions spécifiées ne sont pas refusées. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI

> **Important**
>
> Cette politique ne permet aucune action. Utilisez cette stratégie conjointement à d'autres stratégies qui autorisent des actions spécifiques.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ExplicitDenyIfNotTheOwner",
            "Effect": "Deny",
            "Action": [
                "datapipeline:ActivatePipeline",
                "datapipeline:AddTags",
                "datapipeline:DeactivatePipeline",
                "datapipeline:DeletePipeline",
                "datapipeline:DescribeObjects",
                "datapipeline:EvaluateExpression",
                "datapipeline:GetPipelineDefinition",
                "datapipeline:PollForTask",
                "datapipeline:PutPipelineDefinition",
                "datapipeline:QueryObjects",
                "datapipeline:RemoveTags",
                "datapipeline:ReportTaskProgress",
                "datapipeline:ReportTaskRunnerHeartbeat",
                "datapipeline:SetStatus",
                "datapipeline:SetTaskStatus",
                "datapipeline:ValidatePipelineDefinition"
            ],
            "Resource": ["*"],
            "Condition": {
                "StringNotEquals": {"datapipeline:PipelineCreator": "${aws:userid}"}
            }
        }
    ]
}
```

Amazon DynamoDB : autorise l'accès à une table spécifique

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès complet à la table DynamoDB `MyTable`. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

> **Important**
>
> Cette politique autorise toutes les actions pouvant être effectuées sur une table DynamoDB. Pour vérifier ces actions, consultez Autorisations d'API Amazon DynamoDB : référence des actions, ressources et conditions dans le Guide du développeur Amazon DynamoDB. Vous pouvez fournir les mêmes autorisations en répertoriant chaque action individuelle. Toutefois, si vous utilisez le caractère générique (*) dans l'élément `Action`, par exemple `"dynamodb:List*"`, vous n'avez pas besoin de mettre à jour votre politique si ajoute une nouvelle action Liste.

Cette politique autorise les actions DynamoDB uniquement sur les tables existantes et dont le nom est spécifié. Pour permettre à vos utilisateurs `Read` d'accéder à tous les éléments de DynamoDB, vous pouvez également associer la politique gérée par AmazonDynamola ReadOnlyAccess AWS base de données.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListAndDescribe",
            "Effect": "Allow",
            "Action": [
                "dynamodb:List*",
                "dynamodb:DescribeReservedCapacity*",
                "dynamodb:DescribeLimits",
                "dynamodb:DescribeTimeToLive"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SpecificTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGet*",
                "dynamodb:DescribeStream",
                "dynamodb:DescribeTable",
                "dynamodb:Get*",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchWrite*",
```

```
                "dynamodb:CreateTable",
                "dynamodb:Delete*",
                "dynamodb:Update*",
                "dynamodb:PutItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/MyTable"
        }
    ]
}
```

Amazon DynamoDB : autorise l'accès à des attributs spécifiques

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès aux attributs DynamoDB spécifiques. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

L'exigence `dynamodb:Select` empêche l'action d'API de retourner les attributs qui ne sont pas autorisés, par exemple à partir d'une projection d'index. Pour en savoir plus sur les clés de condition DynamoDB, consultez Spécification de conditions : utilisation de clés de condition dans le Guide du développeur Amazon DynamoDB. Pour en savoir plus sur l'utilisation de plusieurs conditions ou de plusieurs clés de condition dans le `Condition` bloc d'une politique IAM, consultez Plusieurs valeurs dans un élément Condition.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:GetItem",
                "dynamodb:BatchGetItem",
                "dynamodb:Query",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem",
                "dynamodb:DeleteItem",
                "dynamodb:BatchWriteItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/table-name"],
```

```
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:Attributes": [
                        "column-name-1",
                        "column-name-2",
                        "column-name-3"
                    ]
                },
                "StringEqualsIfExists": {"dynamodb:Select": "SPECIFIC_ATTRIBUTES"}
            }
        }
    ]
}
```

Amazon DynamoDB : autorise l'accès au niveau de l'élément à DynamoDB en fonction d'un ID Amazon Cognito

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès au niveau de l'élément à la table DynamoDB `MyTable` en fonction de l'ID d'utilisateur de la réserve d'identités Amazon Cognito. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Pour utiliser cette politique, vous devez structurer votre table DynamoDB pour que l'ID d'utilisateur de la réserve d'identités Amazon Cognito soit la clé de partition. Pour plus d'informations, consultez Création d'une table dans le Guide du développeur Amazon DynamoDB.

Pour en savoir plus sur les clés de condition DynamoDB, consultez Spécification de conditions : utilisation de clés de condition dans le Guide du développeur Amazon DynamoDB.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
```

```
                "dynamodb:Query",
                "dynamodb:UpdateItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/MyTable"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": ["${cognito-identity.amazonaws.com:sub}"]
                }
            }
        }
    ]
}
```

Amazon EC2 : attacher ou détacher des volumes Amazon EBS aux instances EC2 en fonction des balises

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise les propriétaires de volume EBS à attacher ou détacher leurs volumes EBS définis à l'aide de la balise `VolumeUser` aux instances EC2 qui sont balisées en tant qu'instances de développement (`Department=Development`). Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Pour plus d'informations sur la création de politiques IAM pour contrôler l'accès aux ressources Amazon EC2, consultez la section Contrôle de l'accès aux ressources Amazon EC2 dans le guide de l'utilisateur Amazon EC2.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Department": "Development"}
```

```
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:volume/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/VolumeUser": "${aws:username}"}
            }
        }
    ]
}
```

Amazon EC2 : autorise le lancement d'instances EC2 dans un sous-réseau spécifique, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui permet de répertorier les informations de tous les objets EC2 et de lancer des instances EC2 dans un sous-réseau spécifique. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:GetConsole*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/subnet-subnet-id",
```

```
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*::image/ami-*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        }
    ]
}
```

Amazon EC2 : autorise la gestion des groupes de sécurité EC2 avec une paire spécifique d'étiquettes de valeur clé de manière programmatique et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui accorde aux utilisateurs l'autorisation de prendre certaines mesures pour les groupes de sécurité qui ont la même étiquette. Cette politique accorde des autorisations pour afficher les groupes de sécurité dans la console Amazon EC2, ajouter et supprimer des règles entrantes et sortantes, et répertorier et modifier les descriptions de règles pour les groupes de sécurité existants portant l'étiquette `Department=Test`. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeSecurityGroups",
         "ec2:DescribeSecurityGroupRules",
         "ec2:DescribeTags"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:AuthorizeSecurityGroupIngress",
         "ec2:RevokeSecurityGroupIngress",
         "ec2:AuthorizeSecurityGroupEgress",
```

```
            "ec2:RevokeSecurityGroupEgress",
            "ec2:ModifySecurityGroupRules",
            "ec2:UpdateSecurityGroupRuleDescriptionsIngress",
            "ec2:UpdateSecurityGroupRuleDescriptionsEgress"
        ],
        "Resource": [
            "arn:aws:ec2:region:111122223333:security-group/*"
        ],
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/Department": "Test"
            }
        }
      },
      {
        "Effect": "Allow",
        "Action": [
            "ec2:ModifySecurityGroupRules"
        ],
        "Resource": [
            "arn:aws:ec2:region:111122223333:security-group-rule/*"
        ]
      }
    ]
}
```

Amazon EC2 : autorise le démarrage ou l'arrêt des instances EC2 balisées par un utilisateur, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un utilisateur IAM à démarrer ou arrêter des instances EC2, mais uniquement si la balise d'instance `Owner` a la valeur du nom d'utilisateur de cet utilisateur. Cette politique définit des autorisations pour l'accès à la console et par programmation.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
```

```
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Owner": "${aws:username}"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

EC2 : démarrer ou arrêter les instances en fonction des balises

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise le démarrage ou l'arrêt d'instances avec la paire clé-valeur de balise `Project = DataAnalytics`, mais uniquement par les principaux avec la paire clé-valeur de balise `Department = Data`. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

La condition de la politique renvoie la valeur true si les deux parties de la condition sont true. L'instance doit posséder la balise `Project=DataAnalytics`. En outre, le principal IAM (utilisateur ou rôle) qui fait la demande doit posséder la balise `Department=Data`.

> **Note**
>
> À titre de bonne pratique, associez les politiques avec la clé de condition `aws:PrincipalTag` aux groupes IAM pour le cas où certains utilisateurs pourraient avoir la balise spécifiée et d'autres pas.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "StartStopIfTags",
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:region:account-id:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Project": "DataAnalytics",
                    "aws:PrincipalTag/Department": "Data"
                }
            }
        }
    ]
}
```

EC2 : démarrer ou arrêter les instances en fonction des balises de ressources et de principaux correspondantes

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un principal à démarrer ou arrêter une instance Amazon EC2 lorsque la balise de ressource de l'instance et la balise du principal ont la même valeur pour la clé de balise `CostCenter`. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

> **Note**
>
> À titre de bonne pratique, associez les politiques avec la clé de condition `aws:PrincipalTag` aux groupes IAM pour le cas où certains utilisateurs pourraient avoir la balise spécifiée et d'autres pas.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "ec2:startInstances",
            "ec2:stopInstances"
        ],
        "Resource": "*",
        "Condition": {"StringEquals":
            {"aws:ResourceTag/CostCenter": "${aws:PrincipalTag/CostCenter}"}}
    }
}
```

Amazon EC2 : autorise l'accès EC2 complet dans une région spécifique, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un accès EC2 total dans une région spécifique. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique). Pour obtenir la liste des codes de région, veuillez consulter Régions disponibles dans le Guide de l'utilisateur Amazon EC2.

Sinon, vous pouvez utiliser la clé de condition générale `aws:RequestedRegion`, qui est prise en charge par toutes les actions d'API Amazon EC2. Pour de plus amples informations, veuillez consulter la section Exemple : restriction de l'accès à une région spécifique dans le Guide de l'utilisateur d'Amazon EC2.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": "ec2:*",
            "Resource": "*",
            "Effect": "Allow",
            "Condition": {
                "StringEquals": {
                    "ec2:Region": "us-east-2"
                }
            }
        }
```

```
    ]
}
```

Amazon EC2 : autorise le démarrage ou l'arrêt d'une instance EC2 et la modification d'un groupe de sécurité, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise le démarrage ou l'arrêt d'une instance EC2 spécifique, ainsi que la modification d'un groupe de sécurité spécifique. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSecurityGroupReferences",
        "ec2:DescribeStaleSecurityGroups"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "ec2:AuthorizeSecurityGroupEgress",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:RevokeSecurityGroupEgress",
        "ec2:RevokeSecurityGroupIngress",
        "ec2:StartInstances",
        "ec2:StopInstances"
      ],
      "Resource": [
        "arn:aws:ec2:*:*:instance/i-instance-id",
        "arn:aws:ec2:*:*:security-group/sg-security-group-id"
      ],
      "Effect": "Allow"
    }
  ]
```

```
}
```

Amazon EC2 : nécessite MFA (GetSessionToken) pour des opérations EC2 spécifiques

Cet exemple montre comment créer une politique basée sur l'identité qui permet un accès complet à toutes les opérations d' AWS API dans Amazon EC2. Cependant, il refuse explicitement l'accès aux opérations d'API `TerminateInstances` et `StopInstances` si l'utilisateur n'est pas authentifié à l'aide de l'authentification MFA. Pour le faire par programmation, l'utilisateur doit inclure des valeurs `SerialNumber` et `TokenCode` facultatives lors de l'appel de l'opération `GetSessionToken`. Cette opération renvoie les informations d'identification temporaires qui ont été authentifiées à l'aide de MFA. Pour en savoir plus GetSessionToken, consultezGetSessionToken : informations d'identification temporaires pour les utilisateurs dans des environnements non de confiance.

À quoi sert cette politique ?

- L'instruction `AllowAllActionsForEC2` autorise toutes les actions Amazon EC2.
- L'instruction `DenyStopAndTerminateWhenMFAIsNotPresent` rejette les actions `StopInstances` et `TerminateInstances` lorsque le contexte de MFA est manquant. Cela signifie que les actions sont refusées lorsque le contexte d'authentification multi-facteur est manquant (c'est-à-dire, quand l'authentification MFA n'a pas été utilisée). Un refus remplace l'autorisation.

> **Note**
>
> Le contrôle de la condition `MultiFactorAuthPresent` de l'instruction `Deny` ne doit pas être `{"Bool":{"aws:MultiFactorAuthPresent":false}}`, car cette clé n'est pas présente et ne peut pas être évaluée quand l'authentification MFA n'est pas utilisée. Utilisez donc plutôt le contrôle `BoolIfExists` pour voir si la clé est présente avant de vérifier la valeur. Pour plus d'informations, voir ... IfExists opérateurs de conditions.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllActionsForEC2",
```

```
            "Effect": "Allow",
            "Action": "ec2:*",
            "Resource": "*"
        },
        {
            "Sid": "DenyStopAndTerminateWhenMFAIsNotPresent",
            "Effect": "Deny",
            "Action": [
                "ec2:StopInstances",
                "ec2:TerminateInstances"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": false}
            }
        }
    ]
}
```

Amazon EC2 : limite la suspension des instances EC2 à une plage d'adresses IP

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui limite les instances EC2 en permettant l'action, mais en refusant explicitement l'accès lorsque la demande est externe à la plage d'adresses IP spécifiée. Cette politique est utile lorsque les adresses IP de votre entreprise se situent dans les plages d'adresses IP spécifiées. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Si cette politique est utilisée conjointement avec d'autres politiques autorisant l'`ec2:TerminateInstances`action (telles que la politique FullAccess AWS gérée par AmazonEC2), l'accès est refusé. En effet, une instruction de refus explicite est prioritaire dans l'autorisation des instructions. Pour plus d'informations, consultez the section called “Identification d'une demande autorisée ou refusée dans un compte”.

> **Important**
>
> La clé de `aws:SourceIp` condition refuse l'accès à un AWS service AWS CloudFormation, tel que celui qui passe des appels en votre nom. Pour de plus amples informations sur

l'utilisation de la clé de condition `aws:SourceIp`, veuillez consulter AWS clés contextuelles de condition globale.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ec2:TerminateInstances"],
            "Resource": ["*"]
        },
        {
            "Effect": "Deny",
            "Action": ["ec2:TerminateInstances"],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "203.0.113.0/24"
                    ]
                }
            },
            "Resource": ["*"]
        }
    ]
}
```

IAM : Accès à l'API du simulateur de politique

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'utilisation de l'API du simulateur de politique pour les politiques attachées à un utilisateur, groupe d'utilisateurs ou rôle dans l' Compte AWS actuel. Cette politique autorise également d'accéder à la simulation moins sensible des politiques transmises à l'API sous forme de chaînes. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Action": [
                "iam:GetContextKeysForCustomPolicy",
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Pour autoriser un utilisateur à accéder à la console du simulateur de politiques afin de simuler des politiques associées à un utilisateur, à un groupe ou à un rôle dans le courant Compte AWS, consultezIAM : Accès à la console du simulateur de politiques.

IAM : Accès à la console du simulateur de politiques

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'utilisation de la console du simulateur de politique pour les politiques attachées à un utilisateur, groupe d'utilisateurs ou rôle dans l' Compte AWS actuel. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI

Vous pouvez accéder à la console du simulateur de politique IAM à l'adresse suivante :https://policysim.aws.amazon.com/

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetGroup",
                "iam:GetGroupPolicy",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:GetUser",
```

```
                "iam:GetUserPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListAttachedRolePolicies",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroups",
                "iam:ListGroupPolicies",
                "iam:ListGroupsForUser",
                "iam:ListRolePolicies",
                "iam:ListRoles",
                "iam:ListUserPolicies",
                "iam:ListUsers"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

IAM : endosser des rôles qui ont une balise spécifique

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un utilisateur IAM à assumer des rôles avec la paire clé-valeur de balise `Project = ExampleCorpABC`. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Si un rôle avec cette balise existe dans le même compte que l'utilisateur, l'utilisateur peut endosser ce rôle. Si un rôle avec cette balise existe dans un compte autre que celui de l'utilisateur, il a besoin d'autorisations supplémentaires. La politique de confiance du rôle entre comptes doit également autoriser l'utilisateur ou tous les membres du compte de l'utilisateur à endosser le rôle. Pour de plus amples informations sur l'utilisation des rôles pour l'accès entre comptes, veuillez consulter Fournir l'accès à un utilisateur IAM dans un autre utilisateur Compte AWS dont vous êtes le propriétaire.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeTaggedRole",
            "Effect": "Allow",
```

```
            "Action": "sts:AssumeRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:ResourceTag/Project": "ExampleCorpABC"}
            }
        }
    ]
}
```

IAM : autorise et refuse l'accès à plusieurs services par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un accès total à plusieurs services, ainsi qu'un accès autogéré limité dans IAM. Cela refuse également aux utilisateurs l'accès au compartiment des journaux Amazon S3 `logs` ou à l'instance Amazon EC2 `i-1234567890abcdef0` Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

> **Warning**
>
> Cette politique permet un accès total à toutes les actions et ressources dans plusieurs services. Cette politique doit être appliquée uniquement aux administrateurs de confiance.

Vous pouvez utiliser cette politique comme une limite des autorisations pour définir les autorisations maximales qu'une politique basée sur les identités peut accorder à un utilisateur IAM. Pour plus d'informations, veuillez consulter Délégation de responsabilité à d'autres utilisateurs à l'aide des limites d'autorisations. Lorsque la politique est utilisée comme une limite des autorisations pour un utilisateur, les instructions définissent les limites suivantes :

- L'instruction `AllowServices` autorise un accès complet au service AWS spécifié. Cela signifie que les nouvelles actions de l'utilisateur dans ces services sont uniquement limitées par les politiques d'autorisations qui lui sont attachées.
- L'instruction `AllowIAMConsoleForCredentials` autorise l'accès pour répertorier tous les utilisateurs IAM. Cet accès est nécessaire pour accéder à la page Utilisateurs dans la AWS Management Console. Il permet également d'afficher les exigences de mot de passe pour le compte, ce qui est nécessaire pour que l'utilisateur modifie son mot de passe.

- L'instruction `AllowManageOwnPasswordAndAccessKeys` autorise les utilisateurs à gérer uniquement leur propre mot de passe de console et les clés d'accès par programmation. C'est important, car si une autre politique accorde à un utilisateur l'accès complet à IAM, cet utilisateur peut ensuite modifier ses propres autorisations ou celles d'autres utilisateurs. Cette instruction évite ce genre de problème.
- L'instruction `DenyS3Logs` refuse explicitement l'accès au compartiment `logs`. Cette politique applique à l'entreprise des restrictions sur l'utilisateur.
- L'instruction `DenyEC2Production` refuse explicitement l'accès à l'instance `i-1234567890abcdef0`.

Cette politique n'autorise pas l'accès à d'autres services ou actions. Lorsque la politique est utilisée comme limite d'autorisations pour un utilisateur, même si d'autres politiques associées à l'utilisateur autorisent ces actions, AWS refuse la demande.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
```

```
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*LoginProfile*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
}
```

IAM : ajouter une balise spécifique à un utilisateur avec une balise spécifique

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'ajout de la clé de balise `Department` avec les valeurs de balise `Marketing`, `Development` ou `QualityAssurance` à un utilisateur IAM. Cet utilisateur doit déjà inclure la paire clé-valeur de la balise `JobFunction = manager`. Vous pouvez utiliser cette politique pour exiger qu'un gestionnaire appartienne à un seul des trois départements. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

L'instruction `ListTagsForAllUsers` permet d'afficher les balises pour tous les utilisateurs de votre compte.

La première condition de l'instruction `TagManagerWithSpecificDepartment` utilise l'opérateur de condition `StringEquals`. La condition renvoie la valeur true si les deux parties de la condition sont

true. L'utilisateur à baliser doit déjà posséder la balise `JobFunction=Manager`. La demande doit inclure la clé de balise `Department` avec l'une des valeurs de balise répertoriées.

La deuxième condition utilise l'opérateur de condition `ForAllValues:StringEquals`. La condition renvoie la valeur true si toutes les clés de balise dans la demande correspondent à la clé dans la politique. Cela signifie que la seule clé de balise dans la demande doit être `Department`. Pour plus d'informations sur l'utilisation de `ForAllValues`, consultez Clés de contexte à valeurs multiples.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListTagsForAllUsers",
            "Effect": "Allow",
            "Action": [
                "iam:ListUserTags",
                "iam:ListUsers"
            ],
            "Resource": "*"
        },
        {
            "Sid": "TagManagerWithSpecificDepartment",
            "Effect": "Allow",
            "Action": "iam:TagUser",
            "Resource": "*",
            "Condition": {"StringEquals": {
                "iam:ResourceTag/JobFunction": "Manager",
                "aws:RequestTag/Department": [
                    "Marketing",
                    "Development",
                    "QualityAssurance"
                    ]
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "Department"}
            }
        }
    ]
}
```

IAM : ajouter une balise spécifique avec des valeurs spécifiques

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'ajout d'uniquement la clé de balise `CostCenter`, et soit de la valeur de balise `A-123`, soit de la valeur de balise `B-456` à n'importe quel utilisateur ou rôle IAM. Vous pouvez utiliser cette politique pour limiter le balisage à une clé de balise spécifique et de définir les valeurs de balise. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

L'instruction `ConsoleDisplay` permet d'afficher les balises pour tous les utilisateurs et rôles de votre compte.

La première condition de l'instruction `AddTag` utilise l'opérateur de condition `StringEquals`. La condition renvoie la valeur true si la demande inclut la clé de balise `CostCenter` avec l'une des valeurs de balise répertoriées.

La deuxième condition utilise l'opérateur de condition `ForAllValues:StringEquals`. La condition renvoie la valeur true si toutes les clés de balise dans la demande correspondent à la clé dans la politique. Cela signifie que la seule clé de balise dans la demande doit être `CostCenter`. Pour plus d'informations sur l'utilisation de `ForAllValues`, consultez Clés de contexte à valeurs multiples.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ConsoleDisplay",
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:GetUser",
                "iam:ListRoles",
                "iam:ListRoleTags",
                "iam:ListUsers",
                "iam:ListUserTags"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AddTag",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:TagUser",
                "iam:TagRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CostCenter": [
                        "A-123",
                        "B-456"
                    ]
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "CostCenter"}
            }
        }
    ]
}
```

IAM : créer des utilisateurs uniquement avec les balises spécifiques

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise la création d'utilisateurs IAM, mais uniquement avec l'une des clés de balise `Department` et `JobFunction`, ou les deux. La clé de balise `Department` doit avoir la valeur de balise `Development` ou `QualityAssurance`. La clé de balise `JobFunction` doit avoir la valeur de balise `Employee`. Vous pouvez utiliser cette politique pour exiger que les nouveaux utilisateurs aient une fonction et un service spécifiques. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

La première condition de l'instruction utilise l'opérateur de condition `StringEqualsIfExists`. Si une balise avec la clé `JobFunction` ou `Department` est présente dans la demande, la balise doit avoir la valeur spécifiée. Si aucune clé n'est présente, cette condition est évaluée comme true. Le seul moyen selon lequel la condition est évaluée en tant que false est si l'une des clés de condition spécifiées est présente dans la demande, mais possède une valeur différente de celles autorisées. Pour plus d'informations sur l'utilisation de `IfExists`, consultez ... IfExists opérateurs de conditions.

La deuxième condition utilise l'opérateur de condition `ForAllValues:StringEquals`. La condition renvoie la valeur true s'il existe une correspondance entre chacune des clés de balises

spécifiées dans la demande et au moins une valeur dans la politique. Cela signifie que toutes les balises dans la demande doivent être dans cette liste. Toutefois, la demande peut inclure une seule des balises dans la liste. Par exemple, vous pouvez créer un utilisateur IAM avec la balise `Department=QualityAssurance` uniquement. Toutefois, vous ne pouvez pas créer un utilisateur IAM avec les balises `JobFunction=employee` et `Project=core`. Pour plus d'informations sur l'utilisation de `ForAllValues`, consultez Clés de contexte à valeurs multiples.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TagUsersWithOnlyTheseTags",
            "Effect": "Allow",
            "Action": [
                "iam:CreateUser",
                "iam:TagUser"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "aws:RequestTag/Department": [
                        "Development",
                        "QualityAssurance"
                    ],
                    "aws:RequestTag/JobFunction": "Employee"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "Department",
                        "JobFunction"
                    ]
                }
            }
        }
    ]
}
```

IAM : générer et extraire des rapports sur les informations d'identification IAM

Cet exemple montre comment créer une politique basée sur l'identité qui permet aux utilisateurs de générer et de télécharger un rapport répertoriant tous les utilisateurs IAM dans leur Compte AWS Le rapport indique également le statut de plusieurs informations d'identification de l'utilisateur, y compris

les mots de passe, les clés d'accès, les dispositifs MFA et les certificats de signature. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI

Pour de plus amples informations sur les rapports d'informations d'identification, veuillez consulter Obtenir des rapports d'informations d'identification pour votre Compte AWS.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateCredentialReport",
            "iam:GetCredentialReport"
        ],
        "Resource": "*"
    }
}
```

IAM : autorise la gestion des membres d'un groupe par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise la mise à jour de l'appartenance au groupe appelé `MarketingTeam`. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

À quoi sert cette politique ?

- L'instruction `ViewGroups` permet à l'utilisateur de répertorier tous les utilisateurs et groupes dans la AWS Management Console. Elle permet également à l'utilisateur de consulter des informations de base sur les utilisateurs dans le compte. Ces autorisations doivent se trouver dans leur propre instruction, car elles ne prennent pas en charge ou n'ont pas besoin de spécifier d'ARN de ressource particulier. À la place, les autorisations spécifient `"Resource" : "*"`.
- L'instruction `ViewEditThisGroup` permet à l'utilisateur d'afficher des informations sur le groupe `MarketingTeam`, ainsi que d'ajouter et de supprimer des utilisateurs de ce groupe.

Cette politique n'autorise pas l'utilisateur à afficher ou modifier les autorisations des utilisateurs ou du groupe `MarketingTeam`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewGroups",
            "Effect": "Allow",
            "Action": [
                "iam:ListGroups",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:ListGroupsForUser"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ViewEditThisGroup",
            "Effect": "Allow",
            "Action": [
                "iam:AddUserToGroup",
                "iam:RemoveUserFromGroup",
                "iam:GetGroup"
            ],
            "Resource": "arn:aws:iam::*:group/MarketingTeam"
        }
    ]
}
```

IAM : gérer une balise spécifique

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'ajout et la suppression de la balise IAM avec la clé de balise `Department` à des entités IAM (utilisateurs et rôles). Cette politique ne limite pas la valeur de la balise `Department`. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

```
{
```

```
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:TagUser",
            "iam:TagRole",
            "iam:UntagUser",
            "iam:UntagRole"

        ],
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": "Department"}}
    }
}
```

IAM : transmettre un rôle IAM à un service spécifique AWS

Cet exemple montre comment créer une politique basée sur l'identité qui permet de transmettre n'importe quel rôle de service IAM au service Amazon. CloudWatch Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Un rôle de service est un rôle IAM qui indique qu'un AWS service est le principal habilité à assumer le rôle. Cela permet au service d'endosser le rôle et d'accéder aux ressources dans d'autres services en votre nom. Pour permettre CloudWatch à Amazon d'assumer le rôle que vous lui transmettez, vous devez spécifier le principal du `cloudwatch.amazonaws.com` service comme principal dans la politique de confiance de votre rôle. Le principal de service est défini par le service. Pour connaître le principal de service d'un service, consultez la documentation de ce service. Pour certains services, consultez AWS services qui fonctionnent avec IAM et recherchez les services qui contiennent Oui dans la colonne Rôle lié à un service. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service. Recherchez `amazonaws.com` pour afficher le principal de service.

Pour en savoir plus sur la transmission d'un rôle de service à un service, consultez Octroi d'autorisations à un utilisateur pour transférer un rôle à un service AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:PassedToService": "cloudwatch.amazonaws.com"}
            }
        }
    ]
}
```

IAM : autorise l'accès en lecture seule à la console IAM sans reporting

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise des utilisateurs IAM à effectuer toute action IAM dont le nom commence par la chaîne `Get` ou `List`. À mesure que les utilisateurs utilisent la console, celle-ci effectue des demandes à IAM pour répertorier les groupes, les utilisateurs, les rôles et les politiques, et pour générer des rapports sur ces ressources.

L'astérisque agit comme un caractère générique. Lorsque vous utilisez `iam:Get*` dans une politique, les autorisations obtenues incluent toutes les actions IAM commençant par `Get`, telles que `GetUser` et `GetRole`. Les caractères génériques sont utiles si de nouveaux types d'entités sont ajoutés à IAM à l'avenir. Dans ce cas, les autorisations accordées par la politique permettent automatiquement à l'utilisateur de répertorier et d'obtenir les détails relatifs à ces nouvelles entités.

Cette politique ne peut pas servir à générer des rapports ou des derniers détails de service consultés. Pour obtenir une politique différente l'autorisant, veuillez consulter IAM : autorise l'accès en lecture seule à la console IAM.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    }
}
```

IAM : autorise l'accès en lecture seule à la console IAM

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise des utilisateurs IAM à effectuer toute action IAM dont le nom commence par la chaîne `Get`, `List` ou `Generate`. À mesure que les utilisateurs utilisent la console IAM, celle-ci effectue des demandes pour répertorier les groupes, les utilisateurs, les rôles et les politiques, et pour générer des rapports sur ces ressources.

L'astérisque agit comme un caractère générique. Lorsque vous utilisez `iam:Get*` dans une politique, les autorisations obtenues incluent toutes les actions IAM commençant par `Get`, telles que `GetUser` et `GetRole`. L'utilisation d'un caractère générique est avantageuse, en particulier si de nouveaux types d'entités sont ajoutés à IAM à l'avenir. Dans ce cas, les autorisations accordées par la politique permettent automatiquement à l'utilisateur de répertorier et d'obtenir les détails relatifs à ces nouvelles entités.

Utilisez cette politique pour un accès à la console incluant des autorisations pour générer des rapports ou les derniers détails du service consultés. Pour une politique différente n'autorisant pas la génération d'actions, veuillez consulter IAM : autorise l'accès en lecture seule à la console IAM sans reporting.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:Get*",
            "iam:List*",
            "iam:Generate*"
        ],
        "Resource": "*"
    }
}
```

IAM : autorise des utilisateurs spécifiques à gérer un groupe par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise des utilisateurs IAM spécifiques à gérer le groupe `AllUsers`. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de

ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

À quoi sert cette politique ?

- L'instruction `AllowAllUsersToListAllGroups` permet de répertorier tous les groupes. Ceci est nécessaire pour l'accès à la console. Cette autorisation doit se trouver dans sa propre instruction, car elle ne prend pas en charge un ARN de ressource. À la place, les autorisations spécifient `"Resource" : "*"`.
- L'instruction `AllowAllUsersToViewAndManageThisGroup` autorise toutes les actions du groupe pouvant être effectuées sur le type de ressource groupe. Elle n'autorise pas l'action `ListGroupsForUser`, qui peut être effectuée sur un type de ressource d'utilisateur et non pas un type de ressource de groupe. Pour de plus amples informations sur les types de ressources que vous pouvez spécifier pour une action IAM, veuillez consulter Actions, ressources et clés de condition pour AWS Identity and Access Management.
- L'instruction `LimitGroupManagementAccessToSpecificUsers` refuse aux utilisateurs avec les noms spécifiés l'accès aux actions de groupe d'écriture et de gestion des autorisation. Lorsqu'un utilisateur spécifié dans la politique tente d'apporter des modifications au groupe, cette instruction ne permet pas de refuser la demande. Cette demande est autorisée par l'instruction `AllowAllUsersToViewAndManageThisGroup`. Si d'autres utilisateurs tentent d'effectuer ces opérations, la demande est refusée. Vous pouvez afficher les actions IAM définies avec les niveaux d'accès Write (Écriture) ou Permissions management (Gestion des autorisations) lors de la création de cette politique dans la console IAM. Pour ce faire, passez de l'onglet JSON à l'onglet Visual editor (Éditeur visuel). Pour de plus amples informations sur les niveaux d'accès, veuillez consulter Actions, ressources et clés de condition pour AWS Identity and Access Management.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllUsersToListAllGroups",
            "Effect": "Allow",
            "Action": "iam:ListGroups",
            "Resource": "*"
        },
        {
            "Sid": "AllowAllUsersToViewAndManageThisGroup",
            "Effect": "Allow",
```

```
            "Action": "iam:*Group*",
            "Resource": "arn:aws:iam::*:group/AllUsers"
        },
        {
            "Sid": "LimitGroupManagementAccessToSpecificUsers",
            "Effect": "Deny",
            "Action": [
                "iam:AddUserToGroup",
                "iam:CreateGroup",
                "iam:RemoveUserFromGroup",
                "iam:DeleteGroup",
                "iam:AttachGroupPolicy",
                "iam:UpdateGroup",
                "iam:DetachGroupPolicy",
                "iam:DeleteGroupPolicy",
                "iam:PutGroupPolicy"
            ],
            "Resource": "arn:aws:iam::*:group/AllUsers",
            "Condition": {
                "StringNotEquals": {
                    "aws:username": [
                        "srodriguez",
                        "mjackson",
                        "adesai"
                    ]
                }
            }
        }
    ]
}
```

IAM : permet de définir les exigences de mot de passe du compte par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un utilisateur à afficher et mettre à jour ses exigences en matière de mot de passe de compte. Les exigences en matière de mot de passe spécifient les exigences de complexité et les intervalles de rotation obligatoires pour les mots de passe des membres du compte. Cette politique définit des autorisations pour l'accès à la console et par programmation.

Pour savoir comment définir la politique d'exigences de mot de passe de compte pour votre compte, consultez Définition d'une politique de mot de passe du compte pour les utilisateurs IAM.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GetAccountPasswordPolicy",
            "iam:UpdateAccountPasswordPolicy"
        ],
        "Resource": "*"
    }
}
```

IAM : accès à l'API du simulateur de politique en fonction du chemin d'utilisateur

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'utilisation de l'API du simulateur de politique uniquement pour les utilisateurs ayant le chemin `Department/Development`. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
}
```

> **Note**
>
> Pour créer une politique qui autorise l'utilisation de la console du simulateur de politique uniquement pour les utilisateurs ayant un chemin `Department/Development`, consultez IAM : accès à la console du simulateur de politique en fonction du chemin d'utilisateur.

IAM : accès à la console du simulateur de politique en fonction du chemin d'utilisateur

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'utilisation de la console du simulateur de politique uniquement pour les utilisateurs ayant un chemin `Department/Development`. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Vous pouvez accéder au simulateur de politique IAM à l'adresse suivante :https://policysim.aws.amazon.com/

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetPolicy",
                "iam:GetUserPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "iam:GetUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroupsForUser",
                "iam:ListUserPolicies",
                "iam:ListUsers"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
```

```
        }
    ]
}
```

IAM : autorise les utilisateurs IAM à gérer eux-même un dispositif MFA

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise des utilisateurs IAM à autogérer leur périphérique à authentification multifactorielle (MFA). Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI

> **Note**
>
> Si un utilisateur IAM doté de cette politique n'est pas authentifié par MFA, cette politique refuse l'accès à toutes les AWS actions, à l'exception de celles nécessaires pour s'authentifier à l'aide de l'authentification MFA. Si vous ajoutez ces autorisations à un utilisateur connecté à AWS, il peut avoir besoin de se déconnecter puis de se reconnecter pour voir ces modifications.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToCreateVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
```

```
            "Sid": "AllowUserToManageTheirOwnMFA",
            "Effect": "Allow",
            "Action": [
                "iam:EnableMFADevice",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowUserToDeactivateTheirOwnMFAOnlyWhenUsingMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ],
            "Condition": {
                "Bool": {
                    "aws:MultiFactorAuthPresent": "true"
                }
            }
        },
        {
            "Sid": "BlockMostAccessUnlessSignedInWithMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:ListMFADevices",
                "iam:ListUsers",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
```

```
}
```

IAM : Autoriser les utilisateurs IAM à changer leurs informations d'identification par programmation et dans la console

Cet exemple montre comment créer une politique basée sur l'identité qui permet aux utilisateurs IAM de mettre à jour leurs propres clés d'accès, certificats de signature, informations d'identification spécifiques aux services et mots de passe. Cette politique définit des autorisations pour l'accès à la console et par programmation.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        }
    ]
}
```

Pour en savoir plus sur la façon dont les utilisateurs peuvent modifier leur propre mot de passe dans la console, consultez the section called “Modification par l'utilisateur IAM de son propre mot de passe”.

IAM : afficher les dernières informations consultées relatives au service pour une politique Organizations

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'affichage des dernières informations de service consultées pour une politique Organizations spécifique. Cette politique autorise la récupération des données pour la politique de contrôle de service (SCP) avec l'ID `p-policy123`. La personne qui génère et consulte le rapport doit être authentifiée à l'aide AWS Organizations des informations d'identification du compte de gestion. Cette politique autorise le demandeur à récupérer les données de n'importe quelle entité Organizations dans son organisation. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Pour des informations importantes sur les dernières informations consultées, y compris les autorisations requises, la résolution de problèmes et les régions prises en charge, veuillez consulter Affiner les autorisations en AWS utilisant les dernières informations consultées.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOrgsReadOnlyAndIamGetReport",
            "Effect": "Allow",
            "Action": [
                "iam:GetOrganizationsAccessReport",
                "organizations:Describe*",
                "organizations:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowGenerateReportOnlyForThePolicy",
            "Effect": "Allow",
            "Action": "iam:GenerateOrganizationsAccessReport",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:OrganizationsPolicyId": "p-policy123"}
            }
        }
    ]
```

```
}
```

IAM : limite les politiques gérées pouvant être appliquées à un utilisateur, groupe ou rôle IAM

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui limite les politiques AWS gérées par le client et qui peuvent être appliquées à un utilisateur, un groupe ou un rôle IAM. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachUserPolicy",
            "iam:DetachUserPolicy"
        ],
        "Resource": "*",
        "Condition": {
            "ArnEquals": {
                "iam:PolicyARN": [
                    "arn:aws:iam::*:policy/policy-name-1",
                    "arn:aws:iam::*:policy/policy-name-2"
                ]
            }
        }
    }
}
```

AWS: Refusez l'accès aux ressources extérieures à votre compte, à l'exception des AWS politiques IAM gérées

L'utilisation de `aws:ResourceAccount` dans vos politiques basées sur l'identité peut avoir un impact sur la capacité de l'utilisateur ou du rôle à utiliser certains services qui nécessitent une interaction avec des ressources dans des comptes appartenant à un service.

Vous pouvez créer une politique avec une exception pour autoriser les politiques IAM AWS gérées. Un compte géré par un service extérieur à vos AWS Organizations possède des politiques IAM gérées. Il existe quatre actions IAM qui répertorient et récupèrent les AWS politiques gérées. Utilisez ces actions dans l'élément `NotAction` de la déclaration `AllowAccessToS3ResourcesInSpecificAccountsAndSpecificService1` dans la politique.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessToResourcesInSpecificAccountsAndSpecificService1",
      "Effect": "Deny",
      "NotAction": [
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:ListEntitiesForPolicy",
        "iam:ListPolicies"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    }
  ]
}
```

AWS Lambda : autorise une fonction Lambda à accéder à une table Amazon DynamoDB

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès en lecture et en écriture à une table Amazon DynamoDB spécifique. La politique permet également d'écrire des fichiers CloudWatch journaux dans Logs. Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Pour utiliser cette politique, attachez la politique à un rôle de service Lambda. Un rôle de service est un rôle que vous créez dans votre compte pour autoriser un service à effectuer des actions en votre nom. Ce rôle de service doit figurer AWS Lambda en tant que principal dans la politique de confiance. Pour en savoir plus sur l'utilisation de cette politique, consultez Comment créer une politique AWS IAM pour accorder l' AWS Lambda accès à une table AWS Amazon DynamoDB dans le blog sur la sécurité.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGetItem",
                "dynamodb:GetItem",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchWriteItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable"
        },
        {
            "Sid": "GetStreamRecords",
            "Effect": "Allow",
            "Action": "dynamodb:GetRecords",
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable/stream/* "
        },
        {
            "Sid": "WriteLogStreamsAndGroups",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CreateLogGroup",
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
```

```
            "Resource": "*"
        }
    ]
}
```

Amazon RDS : autorise l'accès complet à la base de données RDS dans une région spécifique

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès total à la base de données RDS dans une région spécifique. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "rds:*",
            "Resource": ["arn:aws:rds:region:*:*"]
        },
        {
            "Effect": "Allow",
            "Action": ["rds:Describe*"],
            "Resource": ["*"]
        }
    ]
}
```

Amazon RDS : autorise la restauration des bases de données RDS, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise la restauration de bases de données RDS. Cette politique définit des autorisations pour l'accès à la console et par programmation.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "rds:CreateDBParameterGroup",
                "rds:CreateDBSnapshot",
                "rds:DeleteDBSnapshot",
                "rds:Describe*",
                "rds:DownloadDBLogFilePortion",
                "rds:List*",
                "rds:ModifyDBInstance",
                "rds:ModifyDBParameterGroup",
                "rds:ModifyOptionGroup",
                "rds:RebootDBInstance",
                "rds:RestoreDBInstanceFromDBSnapshot",
                "rds:RestoreDBInstanceToPointInTime"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon RDS : autorise aux propriétaires de balise l'accès complet aux ressource RDS qu'ils ont balisées

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui offre aux propriétaires de balises un accès total aux ressources RDS qu'ils ont balisées. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "rds:Describe*",
                "rds:List*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
```

```
            "Action": [
                "rds:DeleteDBInstance",
                "rds:RebootDBInstance",
                "rds:ModifyDBInstance"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:db-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyOptionGroup",
                "rds:DeleteOptionGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:og-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBParameterGroup",
                "rds:ResetDBParameterGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:pg-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:AuthorizeDBSecurityGroupIngress",
                "rds:RevokeDBSecurityGroupIngress",
                "rds:DeleteDBSecurityGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:secgrp-tag/Owner": "${aws:username}"}
            }
```

```
        },
        {
            "Action": [
                "rds:DeleteDBSnapshot",
                "rds:RestoreDBInstanceFromDBSnapshot"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:snapshot-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBSubnetGroup",
                "rds:DeleteDBSubnetGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:subgrp-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyEventSubscription",
                "rds:AddSourceIdentifierToSubscription",
                "rds:RemoveSourceIdentifierFromSubscription",
                "rds:DeleteEventSubscription"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:es-tag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

Amazon S3 : permet aux utilisateurs Amazon Cognito d'accéder aux objets dans leur compartiment

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise des utilisateurs Amazon Cognito à accéder à des objets dans un compartiment S3 spécifique. Cette politique permet d'accéder uniquement aux objets comportant le mot `cognito`, le nom de l'application, ainsi que l'ID de l'utilisateur fédéré, représenté par la variable ${cognito-identity.amazonaws.com:sub}. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

> **Note**
>
> La valeur « sub » utilisée dans la clé d'objet n'est pas la sous-valeur de l'utilisateur dans le groupe d'utilisateurs, c'est l'ID d'identité associé à l'utilisateur dans le groupe d'identités.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ListYourObjects",
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": [
        "arn:aws:s3:::bucket-name"
      ],
      "Condition": {
        "StringLike": {
          "s3:prefix": [
            "cognito/application-name/${cognito-identity.amazonaws.com:sub}/*"
          ]
        }
      }
    },
    {
      "Sid": "ReadWriteDeleteYourObjects",
      "Effect": "Allow",
```

```
      "Action": [
        "s3:DeleteObject",
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::bucket-name/cognito/application-name/${cognito-
identity.amazonaws.com:sub}/*"
      ]
    }
  ]
}
```

Amazon Cognito assure l'authentification, l'autorisation et la gestion des utilisateurs pour vos applications web et mobiles. Vos utilisateurs peuvent se connecter directement avec un nom d'utilisateur et un mot de passe, ou via un tiers tels que Facebook, Amazon ou Google.

Les deux principaux composants d'Amazon Cognito sont les groupes d'utilisateurs et les groupes d'identités. Les groupes d'utilisateurs sont des répertoires d'utilisateurs qui fournissent des options d'inscription et de connexion pour les utilisateurs de votre application. Les groupes d'identités vous permettent d'accorder à vos utilisateurs l'accès à d'autres AWS services. Vous pouvez utiliser des groupes d'identités et des groupes d'utilisateurs séparément ou conjointement.

Pour plus d'informations sur Amazon Cognito, consultez le Guide de l'utilisateur Amazon Cognito.

Amazon S3 : autorise les utilisateurs fédérés à accéder à leur répertoire de base S3, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise des utilisateurs fédérés à accéder à leur propre objet de compartiment de répertoire de base dans S3. Le répertoire de base est un compartiment qui inclut un dossier `home` et des dossiers pour les utilisateurs fédérés individuels. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

La variable `${aws:userid}` de cette politique se résout à `role-id:specified-name`. La partie `role-id` de l'ID de l'utilisateur fédéré est un identifiant unique attribué au rôle de l'utilisateur fédéré au moment de la création. Pour plus d'informations, consultez Identifiants uniques. `specified-`

nameIl s'agit du RoleSessionName paramètre transmis à la `AssumeRoleWithWebIdentity` demande lorsque l'utilisateur fédéré a assumé son rôle.

Vous pouvez afficher l'ID du rôle à l'aide de la AWS CLI commande`aws iam get-role --role-name` *`specified-name`*. Par exemple, imaginons que vous spécifiez le nom convivial `John` et que l'CLI renvoie l'ID de rôle `AROAXXT2NJT7D3SIQN7Z6`. Dans ce cas, l'ID de l'utilisateur fédéré est `AROAXXT2NJT7D3SIQN7Z6:John`. Cette politique permet ensuite à l'utilisateur fédéré John d'accéder au compartiment Amazon S3 avec le préfixe `AROAXXT2NJT7D3SIQN7Z6:John`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::bucket-name",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:userid}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
```

```
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name/home/${aws:userid}",
                "arn:aws:s3:::bucket-name/home/${aws:userid}/*"
            ]
        }
    ]
}
```

Amazon S3 : accès au compartiment S3, mais compartiment de production refusé sans MFA récente

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise un administrateur Amazon S3 à accéder à n'importe quel compartiment, notamment pour la mise à jour, l'ajout et la suppression d'objets. Cependant, il refuse explicitement l'accès au compartiment `Production` si l'utilisateur ne s'est pas connecté via Multi-Factor Authentication (MFA) au cours des trente dernières minutes. Cette politique accorde les autorisations nécessaires pour effectuer cette action dans la console ou par programmation à l'aide de l'API AWS CLI or AWS . Pour utiliser cette politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Cette politique n'autorise jamais un accès par programme au compartiment `Production` à l'aide des clés d'accès de l'utilisateur à long terme. Ceci est possible en utilisant la clé de condition `aws:MultiFactorAuthAge` avec l'opérateur de condition `NumericGreaterThanIfExists`. Cette condition de politique renvoie `true` si MFA n'est pas présent ou si l'âge de la MFA est supérieur à 30 minutes. Dans ces situations, l'accès est refusé. Pour accéder au `Production` compartiment par programmation, l'administrateur S3 doit utiliser des informations d'identification temporaires qui ont été générées au cours des 30 dernières minutes à l'aide de l'opération GetSessionTokenAPI.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListAllS3Buckets",
            "Effect": "Allow",
            "Action": ["s3:ListAllMyBuckets"],
            "Resource": "arn:aws:s3:::*"
        },
```

```
        {
            "Sid": "AllowBucketLevelActions",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketObjectActions",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:DeleteObject"
            ],
            "Resource": "arn:aws:s3:::*/*"
        },
        {
            "Sid": "RequireMFAForProductionBucket",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::Production/*",
                "arn:aws:s3:::Production"
            ],
            "Condition": {
                "NumericGreaterThanIfExists": {"aws:MultiFactorAuthAge": "1800"}
            }
        }
    ]
}
```

Amazon S3 : autorise les utilisateurs IAM à accéder à leur répertoire de base S3, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise des utilisateurs IAM à accéder à leur propre objet de compartiment de répertoire de base dans S3. Le répertoire d'accueil est un compartiment qui inclut un dossier home et des dossiers pour

les utilisateurs individuels. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Cette politique ne fonctionne pas lors de l'utilisation de rôles IAM, car la variable `aws:username` n'est pas disponible lorsque vous utilisez des rôles IAM. Pour plus d'informations sur les valeurs de clé principale, consultez la section Valeurs de la clé du principal.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::bucket-name",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:username}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
```

```
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name/home/${aws:username}",
                "arn:aws:s3:::bucket-name/home/${aws:username}/*"
            ]
        }
    ]
}
```

Amazon S3 : restreindre la gestion à un compartiment S3 spécifique

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui limite la gestion d'un compartiment Amazon S3 à ce compartiment spécifique. Cette politique accorde l'autorisation d'effectuer toutes les actions Amazon S3, mais refuse l'accès à tous les Service AWS , à l'exception d'Amazon S3. Consultez l'exemple suivant. Selon cette politique, vous ne pouvez accéder qu'aux actions Amazon S3 que vous pouvez effectuer sur un compartiment S3 ou une ressource objet S3. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

Si cette politique est utilisée en combinaison avec d'autres politiques (telles que les politiques FullAccess AWS gérées par AmazonS3 FullAccess ou AmazonEC2) qui autorisent les actions refusées par cette politique, l'accès est refusé. En effet, une instruction de refus explicite est prioritaire dans l'autorisation des instructions. Pour plus d’informations, veuillez consulter the section called “Identification d'une demande autorisée ou refusée dans un compte”.

> **Warning**
>
> `NotAction` et `NotResource` sont des éléments de politique avancée qui doivent être utilisés avec précaution. Cette politique refuse l'accès à chaque service AWS , à l'exception d'Amazon S3. Si vous attachez cette politique à un utilisateur, toutes les autres politiques qui accordent des autorisations aux autres services sont ignorées et l'accès est refusé.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        },
        {
            "Effect": "Deny",
            "NotAction": "s3:*",
            "NotResource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        }
    ]
}
```

Amazon S3 : autorise l'accès en lecture et en écriture aux objets d'un compartiment S3.

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès `Read` et `Write` aux objets d'un compartiment S3 spécifique. Cette politique accorde les autorisations nécessaires pour effectuer cette action par programmation à partir de l' AWS API ou. AWS CLI Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

L'action `s3:*Object` utilise un caractère générique dans le nom de l'action. L'instruction `AllObjectActions` autorise les actions `GetObject`, `DeleteObject`, `PutObject` et toute autre action Amazon S3 qui se termine par le mot « Object ».

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": ["s3:ListBucket"],
            "Resource": ["arn:aws:s3:::bucket-name"]
```

```
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::bucket-name/*"]
        }
    ]
}
```

> **Note**
>
> Pour autoriser l'accès en `Read` et en `Write` aux objets d'un compartiment Amazon S3 spécifique et inclure également des autorisations supplémentaires pour l'accès à la console, consultez Amazon S3 : autorise l'accès en lecture et en écriture aux objets d'un compartiment S3, par programmation et dans la console.

Amazon S3 : autorise l'accès en lecture et en écriture aux objets d'un compartiment S3, par programmation et dans la console

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'accès en `Read` et en `Write` aux objets d'un compartiment S3 spécifique. Cette politique définit des autorisations pour l'accès à la console et par programmation. Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

L'action `s3:*Object` utilise un caractère générique dans le nom de l'action. L'instruction `AllObjectActions` autorise les actions `GetObject`, `DeleteObject`, `PutObject` et toute autre action Amazon S3 qui se termine par le mot « Object ».

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
```

```
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": ["arn:aws:s3:::bucket-name"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::bucket-name/*"]
        }
    ]
}
```

Gestion des politiques IAM

IAM vous fournit les outils pour créer et gérer tous types de politiques IAM (politiques gérées et politiques en ligne). Pour ajouter des autorisations à une identité IAM (utilisateur, groupe ou rôle IAM) vous créez une politique , vous la validez, puis vous l'attachez à l'identité. Vous pouvez attacher plusieurs politiques à une identité, et chaque politique peut contenir plusieurs autorisations.

Consultez les ressources suivantes pour en savoir plus :

- Pour plus d'informations sur les différents types de politiques IAM, consultez Politiques et autorisations dans IAM.
- Pour obtenir des informations générales sur l'utilisation des politiques dans IAM, consultez Gestion de l'accès aux AWS ressources.
- Pour de plus amples informations sur l'évaluation des autorisations lorsque plusieurs politiques sont en vigueur pour une identité IAM donnée, veuillez consulter Logique d'évaluation de politiques.

- Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d'informations, voir IAM et quotas AWS STS.

Rubriques

- Création de politiques IAM
- Validation de politiques IAM
- Générer des politiques basées sur l'activité d'accès
- Test des politiques IAM avec le simulateur de politiques IAM
- Ajout et suppression d'autorisations basées sur l'identité IAM
- Gestion des versions des politiques IAM
- Modification de politiques IAM
- Suppression de politiques IAM
- Affiner les autorisations en AWS utilisant les dernières informations consultées

Création de politiques IAM

Une politique est une entité qui, lorsqu'elle est attachée une identité ou à une ressource, définit les autorisations de cette dernière. Vous pouvez utiliser l' AWS API AWS Management Console AWS CLI, ou pour créer des politiques gérées par le client dans IAM. Les politiques gérées par le client sont des politiques autonomes que vous gérez dans votre propre Compte AWS. Vous pouvez ensuite associer les politiques aux identités (utilisateurs, groupes et rôles) de votre Compte AWS.

Une politique attachée à une identité dans IAM est appelée politique basée sur l'identité. Les politiques basées sur l'identité peuvent inclure des politiques AWS gérées, des politiques gérées par le client et des politiques intégrées. AWS les politiques gérées sont créées et gérées par AWS. Vous pouvez les utiliser, mais vous ne pouvez pas les gérer. Une politique en ligne peut être créée et intégrée directement à un groupe, un utilisateur ou un rôle IAM. Les politiques en ligne ne peuvent pas être réutilisées sur d'autres identités ou gérées en dehors de l'identité où elles existent. Pour plus d’informations, consultez Ajout et suppression d'autorisations basées sur l'identité IAM.

Utilisation de politiques gérées par le client au lieu de politiques en ligne. Il est également préférable d'utiliser des politiques gérées par le client plutôt que des politiques AWS gérées. AWS les politiques gérées fournissent généralement des autorisations administratives ou en lecture seule étendues. Pour plus de sécurité, accordez les privilèges les plus faibles possibles, ce qui consiste à accorder uniquement les autorisations nécessaires à l'exécution de tâches spécifiques.

Lorsque vous créez ou modifiez des politiques IAM, vous AWS pouvez effectuer automatiquement la validation des politiques pour vous aider à créer une politique efficace avec le moindre privilège à l'esprit. Dans le AWS Management Console, IAM identifie les erreurs de syntaxe JSON, tandis qu'IAM Access Analyzer fournit des vérifications de politique supplémentaires avec des recommandations pour vous aider à affiner davantage vos politiques. Pour en savoir plus sur la validation de politiques, veuillez consulter Validation de politiques IAM. Pour en savoir plus sur les vérifications des politiques IAM Access Analyzer et les recommandations exploitables, veuillez consulter Validation de politique IAM Access Analyzer.

Vous pouvez utiliser l' AWS API AWS Management Console AWS CLI, ou pour créer des politiques gérées par le client dans IAM. Pour plus d'informations sur l'utilisation de AWS CloudFormation modèles pour ajouter ou mettre à jour des politiques, consultez la référence aux types de AWS Identity and Access Management ressources dans le Guide de AWS CloudFormation l'utilisateur.

Rubriques

- Création de politiques de rôle IAM (console)
- Création de politiques IAM (AWS CLI)
- Création de politiques IAM (AWS API)

Création de politiques de rôle IAM (console)

Une politique est une entité qui, lorsqu'elle est attachée une identité ou à une ressource, définit les autorisations de cette dernière. Vous pouvez utiliser le AWS Management Console pour créer des politiques gérées par le client dans IAM. Les politiques gérées par le client sont des politiques autonomes que vous gérez dans votre propre Compte AWS. Vous pouvez ensuite associer les politiques aux identités (utilisateurs, groupes et rôles) de votre Compte AWS.

Rubriques

- Création de politiques IAM
- Création de politiques à l'aide de l'éditeur JSON
- Création de politiques avec l'éditeur visuel
- Importation de politiques gérées existantes

Création de politiques IAM

Vous pouvez créer une politique gérée par le client en AWS Management Console utilisant l'une des méthodes suivantes :

- JSON — coller et personnaliser un exemple de politique basée sur l'identité publié.
- Visual editor (Éditeur visuel) — construire intégralement une nouvelle politique dans l'éditeur visuel. Si vous utilisez l'éditeur visuel, vous n'avez pas besoin de comprendre la syntaxe JSON.
- Import (Importation) — importer et personnaliser une politique gérée depuis votre compte. Vous pouvez importer une politique AWS gérée ou une politique gérée par le client que vous avez créée précédemment.

Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d'informations, consultez IAM et quotas AWS STS.

Création de politiques à l'aide de l'éditeur JSON

Vous pouvez taper ou coller des politiques dans JSON à l'aide de l'option JSON. Cette méthode est utile pour copier un exemple de politique à utiliser dans votre compte. Ou vous pouvez composer votre propre document de politique JSON dans l'éditeur JSON. Vous pouvez également utiliser l'option JSON pour basculer entre l'éditeur visuel et JSON afin de comparer les vues.

Lorsque vous créez ou modifiez une politique dans l'éditeur JSON, IAM effectue une validation de politique pour vous aider à créer une politique efficace. IAM identifie les erreurs de syntaxe JSON, tandis qu'IAM Access Analyzer fournit des vérifications de politique supplémentaires avec des recommandations pratiques pour vous aider à affiner la politique.

Un document de politique JSON est composé d'une ou plusieurs instructions. Chaque instruction doit contenir toutes les actions qui partagent le même effet (`Allow` ou `Deny`) et qui prennent en charge les mêmes ressources et conditions. Si une action nécessite que toutes les ressources soient spécifiées (`"*"`) et qu'une autre action prend en charge l'Amazon Resource Name (ARN) d'une ressource spécifique, elles doivent se trouver dans deux instructions JSON distinctes. Pour plus d'informations sur les formats ARN, consultez Amazon Resource Name (ARN) dans le guide Références générales AWS . Pour obtenir des informations d'ordre général sur les politiques IAM, consultez Politiques et autorisations dans IAM. Pour en savoir plus sur le langage de politique IAM, consultez Référence de politique JSON IAM.

Pour utiliser l'éditeur de politique JSON afin de créer une politique

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, choisissez Politiques.
3. Sélectionnez Créer une politique.
4. Dans la section Éditeur de politiques, choisissez l'option JSON.
5. Composez ou collez un document de politique JSON. Pour plus d'informations sur le langage de politique IAM, consultez Référence de politique JSON IAM.
6. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant.

   > **Note**
   >
   > Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez Restructuration de politique.

7. (Facultatif) Lorsque vous créez ou modifiez une politique dans le AWS Management Console, vous pouvez générer un modèle de stratégie JSON ou YAML que vous pouvez utiliser dans les AWS CloudFormation modèles.

   Pour ce faire, dans l'éditeur de politiques, sélectionnez Actions, puis sélectionnez Générer CloudFormation un modèle. Pour en savoir plus, AWS CloudFormation consultez la référence AWS Identity and Access Management aux types de ressources dans le guide de AWS CloudFormation l'utilisateur.
8. Lorsque vous avez fini d'ajouter des autorisations à la politique, choisissez Suivant.
9. Sur la page Vérifier et créer, tapez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vérifiez les Autorisations définies dans cette politique pour voir les autorisations accordées par votre politique.
10. (Facultatif) Ajoutez des métadonnées à la politique en associant les balises sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Balisage des ressources IAM.
11. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

Une fois que vous avez créé une politique, vous pouvez l'attacher à vos utilisateurs, groupes ou rôles. Pour plus d'informations, veuillez consulter Ajout et suppression d'autorisations basées sur l'identité IAM.

Création de politiques avec l'éditeur visuel

L'éditeur visuel de la console IAM vous guide au cours de la création d'une politique sans que vous ayez besoin d'écrire une syntaxe JSON. Pour voir un exemple d'utilisation de l'éditeur visuel pour la création d'un politique, consultez the section called "Contrôle de l'accès aux identités".

Pour utiliser l'éditeur visuel afin de créer une politique

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, choisissez Politiques.
3. Sélectionnez Créer une politique.
4. Dans la section Éditeur de politiques, recherchez la section Sélectionner un service, puis choisissez un AWS service. Vous pouvez utiliser le menu Filtre ou la zone de recherche en haut de l'écran pour limiter les résultats dans la liste des services. Vous pouvez choisir un seul service par bloc d'autorisation de l'éditeur visuel. Pour accorder l'accès à plusieurs services, ajoutez de multiples blocs d'autorisation en sélectionnant Ajouter d'autres autorisations.
5. Dans Actions autorisées, choisissez les actions à ajouter à la politique. Vous pouvez choisir des actions de différentes manières :

   - Activez la case à cocher pour toutes les actions.
   - Choisissez Ajouter des actions pour saisir le nom d'une action spécifique. Vous pouvez utiliser des caractères génériques (*) pour spécifier plusieurs actions.
   - Sélectionnez l'un des groupes de niveau Accès pour choisir toutes les actions pour le niveau d'accès (par exemple, Lecture, Écriture ou Liste).
   - Développez chacun des groupes de Niveaux d'accès pour choisir des actions individuelles.

   Par défaut, la politique que vous créez autorise les actions que vous choisissez. Pour refuser les actions choisies, sélectionnez Switch to deny permissions (Basculer vers le refus des autorisations). Le comportement par défaut d'IAM étant le refus, nous vous recommandons comme bonne pratique de sécurité de n'autoriser un utilisateur à accéder qu'aux actions et aux ressources nécessaires. Vous devez créer une instruction JSON pour refuser des autorisations seulement si vous souhaitez remplacer une autorisation séparément autorisée par une autre

instruction ou politique. Nous vous recommandons de limiter le nombre de refus d'autorisation au minimum, car ils peuvent rendre la résolution des problèmes d'autorisation plus complexe.

6. Pour Ressources, si le service et les actions que vous avez sélectionnés lors des étapes précédentes ne prennent pas en charge le choix de ressources spécifiques, toutes les ressources sont autorisées et vous ne pouvez pas modifier cette section.

   Si vous avez choisi une ou plusieurs actions qui prennent en charge les autorisations de niveau ressource, l'éditeur visuel affiche la liste de ces ressources. Vous pouvez alors développer Ressources pour spécifier les ressources de votre politique.

   Vous pouvez spécifier des ressources de la manière suivante :

   - Choisissez Ajouter des ARN pour spécifier des ressources par leur Amazon Resource Name (ARN). Vous pouvez utiliser l'éditeur visuel ARN ou répertorier les ARN manuellement. Pour de plus amples informations sur la syntaxe ARN, veuillez consulter Amazon Resource Name (ARN) dans le Guide Références générales AWS . Pour plus d'informations sur l'utilisation des ARN dans l'élément `Resource` d'une politique, consultez Éléments de politique JSON IAM : Resource.
   - Choisissez Toute ressource dans ce compte en regard d'une ressource pour accorder des autorisations à toutes les ressources de ce type.
   - Choisissez Toutes pour choisir toutes les ressources pour le service.

7. (Facultatif) Choisissez Demander des conditions – facultatif pour ajouter des conditions à la politique que vous créez. Des conditions limitent l'effet d'une instruction de politique JSON. Par exemple, vous pouvez spécifier qu'un utilisateur est autorisé à effectuer des actions sur les ressources uniquement si la demande de cet utilisateur se produit au cours d'une période spécifiée. Vous pouvez également faire appel à des conditions couramment utilisées pour limiter le fait qu'un utilisateur doive être authentifié à l'aide d'un dispositif MFA (authentification multifacteur). Ou vous pouvez exiger que la demande provienne d'une certaine plage d'adresses IP. Pour obtenir la liste de toutes les clés contextuelles que vous pouvez utiliser dans une condition de politique, consultez la section Actions, ressources et clés de condition pour les AWS services dans la référence d'autorisation de service.

   Vous pouvez choisir des conditions de différentes manières :

   - Utilisez les cases à cocher pour sélectionner les conditions couramment utilisées.
   - Choisissez Ajouter une autre condition pour spécifier d'autres conditions. Choisissez la Clé de condition, le Qualificateur et l'Opérateur de la condition, puis saisissez une Valeur. Pour

ajouter plusieurs valeurs, choisissez Ajouter. Vous pouvez considérer que les valeurs sont connectées par un opérateur logique « OU ». Lorsque vous avez fini, choisissez Ajouter une condition.

Pour ajouter plusieurs conditions, choisissez de nouveau Ajouter une autre condition. Répétez l'opération si nécessaire. Chaque condition s'applique uniquement à ce bloc d'autorisation de l'éditeur visuel. Toutes les conditions doivent être remplies pour que le bloc d'autorisation soit considérée comme réussi. En d'autres termes, considérez les conditions comme étant connectées par un opérateur logique « ET ».

Pour plus d'informations sur l'élément Condition, consultez Éléments de politique JSON IAM : Condition dans le document Référence de politique JSON IAM.

8. Pour ajouter d'autres blocs d'autorisation, choisissez Ajouter d'autres autorisations. Pour chaque bloc, répétez les étapes 2 à 5.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez Restructuration de politique.

9. (Facultatif) Lorsque vous créez ou modifiez une politique dans le AWS Management Console, vous pouvez générer un modèle de stratégie JSON ou YAML que vous pouvez utiliser dans les AWS CloudFormation modèles.

   Pour ce faire, dans l'éditeur de politiques, sélectionnez Actions, puis sélectionnez Générer CloudFormation un modèle. Pour en savoir plus, AWS CloudFormation consultez la référence AWS Identity and Access Management aux types de ressources dans le guide de AWS CloudFormation l'utilisateur.

10. Lorsque vous avez fini d'ajouter des autorisations à la politique, choisissez Suivant.
11. Sur la page Vérifier et créer, tapez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vérifiez les Autorisations définies dans cette politique pour vous assurer que vous les avez accordées comme prévu.

12. (Facultatif) Ajoutez des métadonnées à la politique en associant les balises sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, veuillez consulter Balisage des ressources IAM.
13. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

Une fois que vous avez créé une politique, vous pouvez l'attacher à vos utilisateurs, groupes ou rôles. Pour plus d'informations, veuillez consulter Ajout et suppression d'autorisations basées sur l'identité IAM.

Importation de politiques gérées existantes

Une manière facile de créer une nouvelle politique consiste à importer dans votre compte une politique gérée existante qui possède au moins certains des autorisations dont vous avez besoin. Vous pouvez ensuite personnaliser cette politique pour qu'elle corresponde à vos nouveaux besoins.

Vous ne pouvez pas importer une politique en ligne. Pour en savoir plus sur la différence entre les politiques gérées et les politiques en ligne, consultez Politiques gérées et politiques en ligne.

Pour importer une politique gérée existante dans l'éditeur visuel

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, choisissez Politiques.
3. Sélectionnez Créer une politique.
4. Dans l'Éditeur de politiques, choisissez Visuel, puis, sur le côté droit de la page, choisissez Actions, puis Importer une politique.
5. Dans la fenêtre Importer une politique, choisissez les politiques gérées qui correspondent le mieux à celle que vous voulez inclure dans votre nouvelle politique. Vous pouvez utiliser la zone de recherche en haut de la page pour limiter les résultats dans la liste des politiques.
6. Choisissez Importer une politique.

   Les politiques importées sont ajoutées dans de nouveaux blocs d'autorisation au bas de votre politique.
7. Utilisez Éditeur visuel ou choisissez JSON pour personnaliser votre politique. Ensuite, sélectionnez Suivant.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez Restructuration de politique.

8. Sur la page Vérifier et créer, tapez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vous ne pourrez plus modifier ces paramètres ultérieurement. Vérifiez les Autorisations définies dans cette politique, puis choisissez Créer une politique pour enregistrer votre travail.

Pour importer une politique gérée existante dans l'éditeur JSON

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, choisissez Politiques.
3. Sélectionnez Créer une politique.
4. Dans la section Éditeur de politiques, choisissez l'option JSON, puis, sur le côté droit de la page, choisissez Actions, puis Importer une politique.
5. Dans la fenêtre Importer une politique, choisissez les politiques gérées qui correspondent le mieux à celle que vous voulez inclure dans votre nouvelle politique. Vous pouvez utiliser la zone de recherche en haut de la page pour limiter les résultats dans la liste des politiques.
6. Choisissez Importer une politique.

   Les instructions des politiques importées sont ajoutées au bas de votre politique JSON.
7. Personnalisez votre politique dans JSON. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant. Personnalisez votre politique dans JSON ou sélectionnez Visual editor (Éditeur visuel). Ensuite, sélectionnez Suivant.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur

visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez Restructuration de politique.

8. Sur la page Vérifier et créer, tapez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vous ne pourrez plus modifier ces champs ultérieurement. Vérifiez la politique Autorisations définies dans cette politique, puis choisissez Créer une politique pour enregistrer votre travail.

Une fois que vous avez créé une politique, vous pouvez l'attacher à vos utilisateurs, groupes ou rôles. Pour plus d'informations, voir Ajout et suppression d'autorisations basées sur l'identité IAM.

Création de politiques IAM (AWS CLI)

Une politique est une entité qui, lorsqu'elle est attachée une identité ou à une ressource, définit les autorisations de cette dernière. Vous pouvez utiliser le AWS CLI pour créer des politiques gérées par le client dans IAM. Les politiques gérées par le client sont des politiques autonomes que vous gérez dans votre propre Compte AWS. Selon les bonnes pratiques, nous vous recommandons d'utiliser IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sûres et fonctionnelles. En procédant à la validation de vos politiques, vous pouvez corriger toute erreur ou recommandation avant d'attacher les politiques aux identités (utilisateurs, groupes et rôles) dans votre Compte AWS.

Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d’informations, consultez IAM et quotas AWS STS.

Création de politiques IAM (AWS CLI)

Vous pouvez créer une politique gérée par le client IAM ou une politique en ligne à l'aide de l’interface AWS Command Line Interface (AWS CLI).

Pour créer une politique gérée par le client (AWS CLI)

Utilisez la commande suivante :

- create-policy

Pour créer une politique en ligne pour une identité IAM (groupe, utilisateur ou rôle) (AWS CLI)

Utilisez l’une des commandes suivantes :

- put-group-policy
- put-role-policy
- put-user-policy

> **Note**
>
> Vous ne pouvez pas utiliser IAM pour intégrer une politique en ligne pour un rôle lié à un service.

Pour valider une politique gérée par le client (AWS CLI)

Utilisez la commande IAM Access Analyzer suivante :

- validate-policy

Création de politiques IAM (AWS API)

Une politique est une entité qui, lorsqu'elle est attachée une identité ou à une ressource, définit les autorisations de cette dernière. Vous pouvez utiliser l' AWS API pour créer des politiques gérées par le client dans IAM. Les politiques gérées par le client sont des politiques autonomes que vous gérez dans votre propre Compte AWS. Selon les bonnes pratiques, nous vous recommandons d'utiliser IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sûres et fonctionnelles. En procédant à la validation de vos politiques, vous pouvez corriger toute erreur ou recommandation avant d'attacher les politiques aux identités (utilisateurs, groupes et rôles) dans votre Compte AWS.

Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d’informations, consultez IAM et quotas AWS STS.

Création de politiques IAM (AWS API)

Vous pouvez créer une politique gérée par le client IAM ou une politique en ligne à l'aide de l'API AWS .

Pour créer une politique gérée par le client (AWS API)

Appelez l'opération suivante :

- CreatePolicy

Pour créer une politique en ligne pour une identité IAM (groupe, utilisateur ou rôle) (API AWS)

Appelez l'une des opérations suivantes :

- PutGroupPolicy
- PutRolePolicy
- PutUserPolicy

> **Note**
>
> Vous ne pouvez pas utiliser IAM pour intégrer une politique en ligne pour un rôle lié à un service.

Pour valider une politique gérée par le client (AWS API)

Appelez l'opération IAM Access Analyzer suivante :

- ValidatePolicy

Validation de politiques IAM

Une politique est un document JSON écrit à l'aide de la syntaxe des politiques IAM. Lorsque vous attachez une politique à une entité IAM, telle qu'un utilisateur, un groupe ou un rôle, elle octroie des autorisations à cette entité.

Lorsque vous créez ou modifiez des politiques de contrôle d'accès IAM à l'aide du AWS Management Console, les examine AWS automatiquement pour s'assurer qu'elles sont conformes à la grammaire des politiques IAM. Si AWS détermine qu'une politique n'est pas conforme à la syntaxe, il vous invite à corriger la politique.

IAM Access Analyzer fournit des vérifications de politiques supplémentaires avec des recommandations pour vous aider à affiner davantage la politique. Pour en savoir plus sur les vérifications des politiques IAM Access Analyzer et les recommandations exploitables, veuillez consulter Validation de politique IAM Access Analyzer. Pour afficher la liste des avertissements,

erreurs et suggestions renvoyés par IAM Access Analyzer, veuillez consulter la Référence de vérification de politique IAM Access Analyzer.

Portée de la validation

AWS vérifie la syntaxe et la grammaire des politiques JSON. vérifie également que vos ARN sont correctement formatés et que les noms d'action et les clés de condition sont corrects.

Accès à la validation des politiques

Les politiques sont validées automatiquement lorsque vous créez une politique JSON ou que vous modifiez une politique existante dans la AWS Management Console. Si la syntaxe de la politique n'est pas valide, vous recevez une notification pour résoudre le problème avant de continuer. Les résultats de la validation de la politique d'IAM Access Analyzer sont automatiquement renvoyés dans le fichier AWS Management Console si vous disposez d'autorisations pour. `access-analyzer:ValidatePolicy` Vous pouvez également valider les politiques à l'aide de l' AWS API ou AWS CLI.

Politiques existantes

Il peut arriver que des politiques existantes ne soient pas valides, car leur création ou leur enregistrement sont antérieurs aux dernières mises à jour du moteur de politique. Selon les bonnes pratiques, nous vous recommandons d'utiliser IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sûres et fonctionnelles. Nous vous recommandons d'ouvrir vos politiques existantes et d'examiner les résultats de validation des politiques qui sont générés. Vous ne pouvez pas modifier et enregistrer des politiques existantes sans corriger les erreurs de syntaxe qu'elles contiennent.

Générer des politiques basées sur l'activité d'accès

En tant qu'administrateur ou développeur, vous pouvez octroyer plus d'autorisations à des entités IAM (utilisateurs ou rôles) qu'elle n'en ont besoin. IAM propose plusieurs options pour vous aider à affiner les autorisations que vous octroyez. Une option consiste à générer une politique IAM basée sur une activité d'accès pour une entité. IAM Access Analyzer examine vos AWS CloudTrail journaux et génère un modèle de politique qui contient les autorisations utilisées par l'entité dans la plage de dates spécifiée. Vous pouvez utiliser le modèle pour créer une politique avec des autorisations précises qui accordent uniquement les autorisations requises pour prendre en charge votre cas d'utilisation spécifique.

Par exemple, imaginez que vous êtes un développeur et que votre équipe d'ingénierie a travaillé sur un projet pour créer une nouvelle application. Pour encourager l'expérimentation et permettre à votre équipe de progresser rapidement, vous avez configuré un rôle avec des autorisations étendues pendant le développement de l'application. Maintenant, l'application est prête pour la production. Avant de la lancer dans le compte de production, vous souhaitez identifier et accorder uniquement les autorisations dont le rôle a besoin pour que l'application fonctionne. Cela vous permet de mieux respecter la bonne pratique qui consiste à appliquer le principe du moindre privilège. Vous pouvez générer une politique basée sur l'activité d'accès du rôle que vous avez utilisé pour l'application dans le compte de développement. Vous pouvez ajuster davantage la politique générée, puis l'attacher à une entité de votre compte de production.

Pour en savoir plus sur la génération de politiques IAM Access Analyzer, veuillez consulter IAM Access Analyzer policy generation (Génération de politiques IAM Access Analyzer).

Test des politiques IAM avec le simulateur de politiques IAM

Pour plus d'informations sur la manière d'utiliser les politiques IAM et pourquoi, consultez Politiques et autorisations dans IAM.

Vous pouvez accéder à la console du simulateur de politiques IAM à l'adresse suivante :https://policysim.aws.amazon.com/

> **Important**
>
> Les résultats du simulateur de politiques peuvent différer de ceux de votre AWS environnement réel. Nous vous recommandons de vérifier vos politiques par rapport à votre AWS environnement réel après les avoir testées à l'aide du simulateur de politiques afin de confirmer que vous avez obtenu les résultats souhaités. Pour plus d'informations, consultez Fonctionnement du simulateur de politiques IAM.

Getting Started with the IAM Policy Simulator

Avec le simulateur de politiques IAM, vous pouvez tester et dépanner les politiques basées sur l'identité et les limites d'autorisations IAM. Voici quelques exemples de ce que vous pouvez réaliser avec le simulateur de politiques :

- Testez les politiques basées sur l'identité qui sont attachées aux utilisateurs, groupes d'utilisateurs ou rôles IAM dans votre compte Compte AWS. Si plusieurs politiques sont attachées à l'utilisateur,

au groupe d'utilisateurs ou au rôle, vous pouvez tester toutes les politiques ou sélectionner des politiques individuelles à tester. Vous pouvez tester les actions qui sont autorisées ou refusées par les politiques sélectionnées pour des ressources spécifiques.

- Testez et résolvez les problèmes liés à l'effet des limites d'autorisations sur les entités IAM. Vous ne pouvez simuler qu'une seule limite d'autorisations à la fois.
- Testez les effets sur les utilisateurs IAM des politiques basées sur les ressources qui sont attachées à des ressources AWS telles que des compartiments Amazon S3, des files d'attente Amazon SQS, des rubriques Amazon SNS ou des coffres-forts Amazon S3 Glacier. Pour utiliser une politique basée sur les ressources dans le simulateur de politiques pour des utilisateurs IAM, vous devez inclure la ressource dans la simulation. Vous devez également cocher la case pour inclure la politique de cette ressource dans la simulation.

> **Note**
>
> La simulation de politiques basées sur les ressources n'est pas prise en charge pour les rôles IAM.

- Si vous Compte AWS êtes membre d'une organisation en AWS Organizations, vous pouvez tester l'impact des politiques de contrôle des services (SCP) sur vos politiques basées sur l'identité.

> **Note**
>
> Le simulateur de politiques n'évalue pas les SCP qui comportent des conditions.

- Testez les nouvelles politiques basées sur l'identité qui ne sont pas encore attachées à un utilisateur, un groupe d'utilisateurs ou un rôle en les saisissant ou en les copiant dans le simulateur de politiques. Elles sont utilisées uniquement dans la simulation et en sont pas enregistrées. Vous ne pouvez pas saisir ou copier les politiques basées sur des ressources dans le simulateur de politiques.
- Testez les politiques basées sur l'identité avec les services, actions et ressources sélectionnés. Par exemple, vous pouvez effectuer des tests pour vérifier que votre politique autorise une entité à exécuter les actions `ListAllMyBuckets`, `CreateBucket` et `DeleteBucket` dans le service Amazon S3 sur un compartiment spécifique.
- Simulez des scénarios réels en fournissant des clés de contexte, comme une adresse IP ou une date, qui sont incluses dans les éléments `Condition` des politiques en cours de test.

> **Note**
>
> Le simulateur de politiques ne simule pas les balises fournies comme entrée si la politique basée sur l'identité de la simulation ne comporte aucun élément `Condition` qui vérifie explicitement la présence de balises.

- Identifiez quelle instruction spécifique d'une politique basée sur l'identité entraîne l'autorisation ou le refus de l'accès à une ressource ou une action particulière.

Rubriques

- Fonctionnement du simulateur de politiques IAM
- Autorisations nécessaires pour utiliser le simulateur de politiques IAM
- Utilisation du simulateur de politique IAM (Console)
- Utilisation du simulateur de politique IAM (AWS CLI et de AWS l'API)

Fonctionnement du simulateur de politiques IAM

Le simulateur de politiques évalue les déclarations de la politique basée sur l'identité et les entrées que vous fournissez lors de la simulation. Les résultats du simulateur de politiques peuvent différer de ceux de votre environnement AWS en ligne. Nous vous recommandons de vérifier vos politiques par rapport à votre AWS environnement réel après les avoir testées à l'aide du simulateur de politiques afin de confirmer que vous avez obtenu les résultats souhaités.

Le simulateur de politiques se distingue de l' AWS environnement réel sur les points suivants :

- Le simulateur de politiques n'émet pas de véritable demande de AWS service. Vous pouvez donc tester en toute sécurité les demandes susceptibles d'apporter des modifications indésirables à votre AWS environnement réel. Le simulateur de politiques ne prend pas en compte les valeurs clés du contexte réel dans la production.
- Du fait que le simulateur de politiques ne simule pas l'exécution des actions sélectionnées, il ne peut rapporter aucune réponse à la demande simulée. Le seul résultat renvoyé est si l'action demandée serait autorisée ou refusée.
- Si vous modifiez une politique dans le simulateur de politiques, ces modifications affectent uniquement le simulateur de politiques. La politique correspondante dans votre dossier Compte AWS demeure inchangée.

- Vous ne pouvez pas tester des politiques de contrôle des services (SCP) avec n'importe quelles conditions.
- Le simulateur de politiques ne prend pas en charge la simulation des rôles et utilisateurs IAM pour l'accès intercompte.

> **Note**
>
> Le simulateur de politiques IAM ne détermine pas quels services prennent en charge les clés de condition globales pour l'autorisation. Par exemple, le simulateur de politiques n'identifie pas les services qui ne prennent pas en charge `aws:TagKeys`.

Autorisations nécessaires pour utiliser le simulateur de politiques IAM

Vous pouvez utiliser la console du simulateur de politiques ou l'API du simulateur de politiques pour tester des politiques. Par défaut, les utilisateurs de la console peuvent tester les politiques qui ne sont pas encore attachées à un utilisateur, un groupe d'utilisateurs ou à un rôle en les saisissant ou en les copiant dans le simulateur de politiques. Ces politiques sont utilisées uniquement dans la simulation et ne divulguent aucune information sensible. Les utilisateurs d'API doivent avoir les autorisations pour tester les politiques détachées. Vous pouvez autoriser les utilisateurs de la console ou de l'API à tester des politiques attachées à des utilisateurs, des groupes d'utilisateurs ou des rôles IAM dans votre Compte AWS. Pour ce faire, vous devez fournir l'autorisation d'extraire ces politiques. Afin de tester les politiques basées sur les ressources, les utilisateurs doivent avoir l'autorisation d'extraire la politique de la ressource.

Pour obtenir des exemples de stratégie de console et d'API qui autorisent l'utilisateur de la console à simuler les politiques, consultez the section called “Exemples de politiques : AWS Identity and Access Management (IAM)”.

Autorisations nécessaires pour utiliser la console du simulateur de politique

Vous pouvez autoriser les utilisateurs à tester des politiques attachées à des utilisateurs, des groupes d'utilisateurs ou des rôles IAM dans votre Compte AWS. Pour ce faire, vous devez fournir à vos utilisateurs les autorisations nécessaires pour extraire ces politiques. Afin de tester les politiques basées sur les ressources, les utilisateurs doivent avoir l'autorisation d'extraire la politique de la ressource.

Pour visualiser un exemple de politique qui autorise l'utilisation de la console du simulateur de politique pour les politiques attachées à un utilisateur, à un groupe d'utilisateurs ou à un rôle, consultez IAM : Accès à la console du simulateur de politiques.

Pour visualiser un exemple de politique qui autorise l'utilisation de la console du simulateur de politique uniquement pour ces utilisateurs avec un chemin spécifique, consultez IAM : accès à la console du simulateur de politique en fonction du chemin d'utilisateur.

Pour créer une politique permettant l'utilisation de la console du simulateur de politique pour un seul type d'entité, utilisez les procédures suivantes.

Pour autoriser les utilisateurs de la console à simuler des politiques pour des utilisateurs

Incluez les actions suivantes dans votre politique :

- `iam:GetGroupPolicy`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetUser`
- `iam:GetUserPolicy`
- `iam:ListAttachedUserPolicies`
- `iam:ListGroupsForUser`
- `iam:ListGroupPolicies`
- `iam:ListUserPolicies`
- `iam:ListUsers`

Pour autoriser les utilisateurs de la console à simuler des politiques pour des groupes d'utilisateurs

Incluez les actions suivantes dans votre politique :

- `iam:GetGroup`
- `iam:GetGroupPolicy`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:ListAttachedGroupPolicies`
- `iam:ListGroupPolicies`

- `iam:ListGroups`

Pour autoriser les utilisateurs de la console à simuler des politiques pour des rôles

Incluez les actions suivantes dans votre politique :

- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetRole`
- `iam:GetRolePolicy`
- `iam:ListAttachedRolePolicies`
- `iam:ListRolePolicies`
- `iam:ListRoles`

Pour tester les politiques basées sur les ressources, les utilisateurs doivent avoir l'autorisation d'extraire la politique de la ressource.

Pour autoriser les utilisateurs de la console à tester les politiques basées sur les ressources dans un compartiment Amazon S3

Incluez l'action suivante dans votre politique :

- `s3:GetBucketPolicy`

Par exemple, la politique suivante utilise cette action pour autoriser les utilisateurs de la console à simuler une politique basée sur une ressource, dans un compartiment Amazon S3 spécifique.

```
{
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": "s3:GetBucketPolicy",
            "Resource":"arn:aws:s3:::bucket-name/*"
          }
        ]
      }
```

Autorisations nécessaires pour utiliser l'API du simulateur de politique

L'API du simulateur de politiques GetContextKeyForCustomPolicyfonctionne et vous SimulateCustomPolicypermet de tester des politiques qui ne sont pas encore associées à un utilisateur, à un groupe d'utilisateurs ou à un rôle. Pour tester des politiques de ce type, vous transmettez les politiques sous forme de chaînes à l'API. Ces politiques sont utilisées uniquement dans la simulation et ne divulguent aucune information sensible. Vous pouvez également utiliser l'API pour tester les politiques attachées à des utilisateurs, des groupes d'utilisateurs ou des rôles IAM dans votre Compte AWS. Pour ce faire, vous devez autoriser les utilisateurs à appeler GetContextKeyForPrincipalPolicyet SimulatePrincipalPolicy.

Pour consulter un exemple de politique qui permet d'utiliser l'API du simulateur de politiques pour les politiques attachées et non attachées dans la version actuelle Compte AWS, consultezIAM : Accès à l'API du simulateur de politique.

Pour créer une politique permettant l'utilisation de l'API du simulateur de politique pour un seul type de politique, utilisez les procédures suivantes.

Pour autoriser les utilisateurs d'API à simuler les politiques transmises directement à l'API sous forme de chaînes

Incluez les actions suivantes dans votre politique :

- `iam:GetContextKeysForCustomPolicy`
- `iam:SimulateCustomPolicy`

Pour autoriser les utilisateurs API à simuler les politiques attachées à des utilisateurs, groupes, rôles ou ressources IAM

Incluez les actions suivantes dans votre politique :

- `iam:GetContextKeysForPrincipalPolicy`
- `iam:SimulatePrincipalPolicy`

Par exemple, pour autoriser un utilisateur nommé Bob à simuler une politique qui est attribuée à un utilisateur nommé Alice, autorisez Bob à accéder aux ressources suivantes : `arn:aws:iam::777788889999:user/alice`.

Pour visualiser un exemple de politique qui autorise l'utilisation de l'API du simulateur de politique uniquement pour ces utilisateurs avec un chemin spécifique, consultez IAM : accès à l'API du simulateur de politique en fonction du chemin d'utilisateur.

Utilisation du simulateur de politique IAM (Console)

Par défaut, les utilisateurs peuvent tester les politiques qui ne sont pas encore attachées à un utilisateur, un groupe d'utilisateurs ou à un, rôle en les saisissant ou en les copiant dans la console du simulateur de politique. Ces politiques sont utilisées uniquement dans la simulation et ne divulguent aucune information sensible.

Pour tester une politique qui n'est pas attachée à un utilisateur, un groupe d'utilisateurs ou un rôle (console)

1. Ouvrez la console du simulateur de politique IAM à l'adresse suivante : https://policysim.aws.amazon.com/.
2. Dans le menu Mode: en haut de la page, sélectionnez New Policy (Nouvelle politique).
3. Dans le menu Policy Sandbox (Environnement de test (sandbox) de politique), choisissez Create New Policy (Créer une politique).
4. Tapez ou copiez une politique dans le simulateur de politiques et utilisez le simulateur de politiques comme décrit dans les étapes suivantes.

Après avoir été autorisé à utiliser la console du simulateur de politiques IAM, vous pouvez utiliser le simulateur de politiques pour tester un utilisateur, un groupe d'utilisateurs, un rôle ou une politique de ressource IAM.

Pour tester une politique attachée à un utilisateur, un groupe d'utilisateurs ou un rôle (console)

1. Ouvrez la console du simulateur de politique IAM à l'adresse suivante : https://policysim.aws.amazon.com/.

> **Note**
>
> Pour vous connecter au simulateur de politique en tant qu'utilisateur IAM, utilisez votre URL de connexion unique pour vous connecter à la AWS Management Console. Ensuite, accédez à https://policysim.aws.amazon.com/. Pour plus d'informations sur la connexion

en tant qu'utilisateur IAM, consultez Comment les utilisateurs d'IAM se connectent à AWS.

Le simulateur de politiques s'ouvre au mode Existing Policies (Politiques existantes) et répertorie les utilisateurs IAM de votre compte sous Users, Groups, and Roles (Utilisateurs, groupes et rôles).

2. Choisissez l'option appropriée à votre tâche :

Pour tester ceci :	Faites ceci :
Une politique attachée à un utilisateur	Choisissez Users (Utilisateurs) dans la liste Users, Groups, and Roles (Utilisateurs, groupes et rôles). Puis choisissez l'utilisateur.
Une politique attachée à un groupe d'utilisateurs	Choisissez Groups (Groupes) dans la liste Users, Groups, and Roles (Utilisateurs, groupes et rôles). Puis sélectionnez le groupe d'utilisa teurs.
Une politique attachée à un rôle	Choisissez Roles (Rôles) dans la liste Users, Groups, and Roles (Utilisateurs, groupes et rôles). Puis choisissez le rôle.
Une politique attachée à une ressource	veuillez consulter Step 9.
Politique personnal isée pour un utilisate ur, un groupe d'utilisateurs ou un rôle	Choisissez Create new policy (Créer une nouvelle politique). Dans le nouveau panneau Policies (Politiques), saisissez ou collez une politique, puis choisissez Apply (Appliquer).

Conseil

Pour tester une politique attachée à un groupe, vous pouvez lancer le simulateur de politiques IAM directement dans la console IAM : dans le panneau de navigation,

sélectionnez Groupes. Choisissez le nom du groupe sur lequel vous souhaitez tester une politique, puis choisissez l'onglet Autorisations. Choisissez Simulate (Simuler).
Pour tester une politique gérée par le client attachée à un utilisateur : dans le panneau de navigation, choisissez Utilisateurs. Choisissez le nom de l'utilisateur dont vous souhaitez modifier les autorisations. Ensuite, choisissez l'onglet Autorisations et développez la politique à tester. À l'extrême droite, choisissez Simuler la politique. Le simulateur de politique IAM ouvre une nouvelle fenêtre et affiche la politique sélectionnée dans le panneau Policies (Politiques).

3. (Facultatif) Si votre compte est membre d'une organisation dans AWS Organizations, activez la case à cocher en regard des SCP AWS Organizations afin d'inclure les SCP dans votre évaluation simulée. Les stratégies de contrôle de service qui spécifient les autorisations maximales pour une organisation ou une unité d'organisation. La stratégie de contrôle de service limite les autorisations pour les entités dans les comptes membres. Si une stratégie de contrôle de service bloque un service ou une action, aucune entité de ce compte ne peut accéder à ce service ni effectuer cette action. Cette affirmation se confirme même si un administrateur accorde explicitement des autorisations à ce service ou à cette action IAM ou par le biais d'un IAM ou d'une politique de ressources.

   Si votre compte n'est pas membre d'une organisation, la case à cocher n'apparaît pas.

4. (Facultatif) Vous pouvez tester une politique définie en tant que limite d'autorisations pour une entité IAM (utilisateur ou rôle), mais pas pour des groupes d'utilisateurs. Si une politique de limite des autorisations est actuellement définie pour l'entité, elle apparaît dans le panneau Policies (Politiques). Vous ne pouvez définir qu'une seule limite d'autorisations pour une entité. Pour tester une limite d'autorisations différente, vous pouvez créer une limite d'autorisations personnalisée. Pour ce faire, choisissez Create New Policy (Créer une nouvelle politique).
   Un nouveau panneau Policies (Politiques) s'ouvre. Dans le menu, choisissez Custom IAM Permissions Boundary Policy (Politique de limite d'autorisations IAM personnalisée). Entrez un nom pour la nouvelle politique et saisissez ou copiez une politique dans l'espace ci-dessous. Choisissez Apply (Appliquer) pour enregistrer la politique. Ensuite, choisissez Back (Précédent) pour revenir au panneau Policies (Politiques) d'origine. Sélectionnez ensuite la case à cocher en regard de la limite d'autorisations que vous souhaitez utiliser pour la simulation.

5. (Facultatif) Vous pouvez tester uniquement un sous-ensemble de politiques attachées à un utilisateur, un groupe d'utilisateurs ou un rôle. Pour ce faire, dans le panneau Policies (Politiques), désactivez la case à cocher en regard de chaque politique que vous souhaitez exclure.

6. Sous Simulateur de politique, choisissez Select service (Sélectionner un service), puis choisissez le service à tester. Choisissez ensuite Sélectionner des actions et sélectionnez une ou plusieurs actions à tester. Bien que les menus affichent les sélections disponibles pour un seul service à la fois, tous les services et actions que vous avez sélectionnés s'affichent dans Action Settings and Results (Paramètres d'action et résultats).
7. (Facultatif) Si des politiques que vous choisissez dans les interfaces Step 2 et Step 5 incluent des conditions avec des clés de condition globale AWS, fournissez des valeurs pour ces clés. Cette opération est possible en étendant la section Global Settings (Paramètres généraux) et en saisissant des valeurs pour les noms de clés affichés ici.

   > **Warning**
   >
   > Si vous laissez la valeur d'une clé de condition vide, cette clé est ignorée pendant la simulation. Dans certains cas, cela entraîne une erreur et l'exécution de la simulation échoue. Dans d'autres cas, la simulation fonctionne, mais les résultats peuvent ne pas être fiables. Dans ces cas, la simulation ne correspond pas aux conditions du monde réel qui incluent une valeur pour la clé de condition ou la variable.

8. (Facultatif) Chaque action sélectionnée s'affiche dans la liste Action Settings and Results (Paramètres d'action et résultats) avec la mention Not simulated (Non simulée) affichée dans la colonne Autorisation jusqu'à ce que vous exécutiez réellement la simulation. Avant d'exécuter la simulation, vous pouvez configurer chaque action avec une ressource. Pour configurer des actions individuelles pour un scénario spécifique, choisissez la flèche pour développer la ligne de l'action. Si l'action prend en charge les autorisations de niveau de ressource, vous pouvez saisir l'Amazon Resource Name (ARN) de la ressource spécifique dont vous souhaitez tester l'accès. Par défaut, chaque ressource est définie sur un caractère générique (*). Vous pouvez également spécifier une valeur pour toutes les clés de contexte de condition. Comme mentionné précédemment, les clés avec des valeurs vides sont ignorées, ce qui peut entraîner des échecs de simulation ou des résultats peu fiables.

   a. Choisissez la flèche située en regard du nom de l'action pour développer chaque ligne et configurer les informations supplémentaires requises pour simuler précisément l'action dans votre scénario. Si l'action nécessite des autorisations de niveau de ressource, vous pouvez saisir l'Amazon Resource Name (ARN) de la ressource spécifique à laquelle vous souhaitez simuler l'accès. Par défaut, chaque ressource est définie sur un caractère générique (*).

b. Si l'action prend en charge les autorisations de niveau ressource mais qu'elles ne sont pas obligatoires, vous pouvez choisir l'option Ajouter une ressource pour sélectionner le type de ressource que vous souhaitez ajouter à la simulation.

c. Si l'une des politiques sélectionnées inclue un élément `Condition` qui fait référence à une clé de contexte pour le service de cette action, ce nom de clé s'affiche sous l'action. Vous pouvez spécifier la valeur à utiliser pendant la simulation de cette action pour la ressource spécifiée.

Actions nécessitant différents groupes de types de ressources

Certaines actions nécessitent différents types de ressources dans certaines circonstances. Chaque groupe de types de ressources est associé à un scénario. Si l'un d'entre eux s'applique à votre simulation, sélectionnez-le et le simulateur de politiques nécessite les types de ressources adaptés à ce scénario. La liste suivante affiche chacune des options de scénarios prises en charge et les ressources que vous devez définir pour exécuter la simulation.

Chacun des scénarios suivants Amazon EC2 nécessite que vous spécifiiez les ressources `instance`, `image` et `security-group`. Si votre scénario inclut un volume EBS, vous devez spécifier ce `volume` en tant que ressource. Si le scénario Amazon EC2 inclut un Virtual Private Cloud (VPC), vous devez fournir la ressource `network-interface`. S'il inclut un sous-réseau IP, vous devez spécifier la ressource `subnet`. Pour de plus amples informations sur les options de scénarios Amazon EC2, veuillez consulter Supported Platforms (Plateformes prises en charge) dans le Guide de l'utilisateur Amazon EC2.

- EC2-VPC- InstanceStore

  instance, image, groupe de sécurité, interface réseau

- Sous-réseau EC2-VPC- InstanceStore

  instance, image, groupe de sécurité, interface réseau, sous-réseau

- EC2-VPC-EBS

  instance, image, groupe de sécurité, interface réseau, volume

- EC2-VPC-EBS-Subnet

  instance, image, groupe de sécurité, interface réseau, sous-réseau, volume

9. (Facultatif) Si vous souhaitez inclure une politique basée sur les ressources dans votre simulation, vous devez d'abord sélectionner les actions que vous souhaitez simuler sur cette ressource dans Step 6. Développez les lignes des actions sélectionnées et saisissez l'ARN de la ressource avec une politique que vous souhaitez simuler. Sélectionnez ensuite Include Resource Policy (Inclure une politique de ressource) en regard de la zone de texte ARN. Le simulateur de politique IAM prend actuellement en charge les politiques basées sur des ressources uniquement dans les services suivants : Amazon S3 (politiques basées sur les ressources uniquement ; les listes de contrôle d'accès (ACL) ne sont pas prises en charge actuellement), Amazon SQS, Amazon SNS et coffres S3 Glacier déverrouillés (les coffres verrouillés ne sont pas pris en charge actuellement).
10. Choisissez Run Simulation (Exécuter la simulation) dans le coin supérieur droit.

    La colonne Autorisation de chaque ligne de Action Settings and Results (Paramètres d'action et résultats) affiche le résultat de la simulation de cette action sur la ressource spécifiée.
11. Pour voir quelle instruction d'une politique a autorisé ou refusé explicitement une action, choisissez le lien ***N*** matching statement(s) (N instruction(s) correspondante(s)) dans la colonne Autorisations pour développer la ligne. Ensuite, affichez le lien Show statement (Afficher l'instruction). Le panneau Politiques affiche la politique correspondant à l'instruction ayant affecté le résultat de la simulation surlignée.

    > **Note**
    >
    > Si une action est implicitement refusée, autrement dit si l'action est refusée uniquement parce qu'elle n'est pas explicitement autorisée, les options List (Répertorier) et Show statement (Afficher l'instruction) ne sont pas affichées.

Dépannage des messages de la console du simulateur de politique IAM

Le tableau suivant répertorie les messages d'information et d'avertissement que vous êtes susceptible de recevoir lorsque vous utilisez le simulateur de politique IAM. Le tableau fournit également des mesures à prendre pour les résoudre.

Message	Mesures à prendre
This policy has been edited. Changes will not be saved to your account.	Aucune action requise.

Message	Mesures à prendre
	Ce message est informatif. Si vous modifiez une politique existante dans le simulateur de politique IAM, votre modification n'affecte pas votre Compte AWS. Le simulateur de politiques vous permet de modifier vos politique s uniquement à des fins de test.
Cannot get the resource policy. Raison : *message d'erreur détaillé*	Le simulateur de politiques ne parvient pas à accéder à une politique basée sur des ressources demandée. Vérifiez que l'ARN de la ressources spécifiée est correct et que l'utilisa teur exécutant la simulation est autorisé à lire la politique de la ressource.
One or more policies require values in the simulation settings. The simulation might fail without these values.	Ce message s'affiche si la politique que vous testez contient des clés de condition ou des variables mais que vous n'avez pas fourni de valeurs pour ces clés ou variables dans Simulation Settings (Paramètres de simulation). Pour ignorer ce message, choisissez Simulatio n Settings (Paramètres de simulation), puis saisissez une valeur pour chaque clé ou variable de condition.
You have changed policies. These results are no longer valid.	Ce message s'affiche si vous avez modifié la politique sélectionnée alors que les résultats sont affichés dans le panneau Résultats. Les résultats affichés dans le panneau Résultats ne sont pas mis à jour de manière dynamique. Pour ignorer ce message, choisissez de nouveau Run Simulation (Exécuter la simulatio n) pour afficher les nouveaux résultats de la simulation basés sur les modifications apportées dans le panneau Politiques.

Message	Mesures à prendre
La ressource que vous avez tapée pour cette simulation ne correspond pas à ce service.	Ce message s'affiche si vous avez tapé un Amazon Resource Name (ARN) dans le panneau Simulation Settings (Paramètres de simulation) qui ne correspond pas au service que vous avez choisi pour la simulation actuelle. Par exemple, ce message s'affiche si vous spécifiez un ARN pour une ressource Amazon DynamoDB, mais que vous avez choisi Amazon Redshift comme service à simuler. Pour ignorer ce message, procédez comme suit : • Supprimez l'ARN de la zone du panneau Simulation Settings (Paramètres de simulation). • Choisissez le service correspondant à l'ARN que vous avez spécifié dans Simulation Settings (Paramètres de simulation).
Cette action appartient à un service qui prend en charge des mécanismes de contrôle d'accès spéciaux, en plus des politiques basées sur les ressources, telles que les listes ACL Amazon S3 ou les politiques de verrouillage de coffre S3 Glacier. The policy simulator does not support these mechanisms, so the results can differ from your production environment.	Aucune action requise. Ce message est informatif. Dans la version actuelle, le simulateur de politiques évalue les politiques attachées aux utilisateurs et aux groupes d'utilisateurs, et peut évaluer les politiques basées sur les ressources pour Amazon S3, Amazon SQS, Amazon SNS et S3 Glacier. Le simulateur de politique ne prend pas en charge tous les mécanismes de contrôle d'accès pris en charge par d'autres services AWS .

Message	Mesures à prendre
DynamoDB FGAC is currently not supported.	Aucune action requise. Ce message d'information fait référence à un contrôle précis des accès. Le contrôle d'accès affiné est la possibilité d'utiliser des conditions de politique IAM pour déterminer qui peut accéder aux éléments de données et attributs individuels dans les tables et les index DynamoDB. Il fait également référence aux actions qui peuvent être effectuées sur ces tables et index. La version actuelle du simulateur de politique IAM ne prend pas en charge ce type de condition de politique. Pour plus d'informations sur le contrôle d'accès précis à DynamoDB, consultez Contrôle d'accès précis pour DynamoDB.
You have policies that do not comply with the policy syntax. Vous pouvez utiliser la validation de politique pour examiner les mises à jour recommandées de vos politiques.	Ce message apparaît en haut de la liste des politiques si vous disposez de politiques non conformes à la syntaxe des politiques IAM. Pour simuler ces politiques, veuillez consulter les options de validation de politiques à Validation de politiques IAM afin d'identifier et corriger ces politiques.
This policy must be updated to comply with the latest policy syntax rules.	Ce message s'affiche si vous disposez de politiques non conformes à la syntaxe des politiques IAM. Pour simuler ces politiques, veuillez consulter les options de validation de politiques à Validation de politiques IAM afin d'identifier et corriger ces politiques.

Utilisation du simulateur de politique IAM (AWS CLI et de AWS l'API)

Les commandes du simulateur de politique nécessitent généralement d'appeler des opérations d'API pour faire deux choses :

1. Évaluer les politiques et renvoyer la liste des clés de contexte auxquelles elles font référence. Vous devez savoir quelles clés de contexte sont référencées pour pouvoir leur attribuer des valeurs à l'étape suivante.
2. Simuler les politiques, en fournissant une liste d'actions, de ressources et de clés de contexte utilisées pendant la simulation.

Pour des raisons de sécurité, les opérations d'API ont été réparties en deux groupes :

- Les opérations API qui simulent uniquement les politiques transmises directement à l'API sous forme de chaînes. Cet ensemble comprend GetContextKeysForCustomPolicyet SimulateCustomPolicy.
- Les opérations API qui simulent les politiques attachées à un utilisateur IAM, un groupe, un rôle ou une ressource spécifiée. Du fait que les opérations d'API peuvent révéler des détails sur les autorisations affectées à d'autres entités IAM, vous devez envisager de restreindre l'accès à ces opérations API. Cet ensemble comprend GetContextKeysForPrincipalPolicyet SimulatePrincipalPolicy. Pour en savoir plus sur les restrictions appliquées aux opérations API, consultez Exemples de politiques : AWS Identity and Access Management (IAM).

Dans les deux cas, les opérations API simulent l'effet d'une ou de plusieurs politiques sur une liste d'actions et de ressources. Chaque action est appariée à chaque ressource et la simulation détermine si les politiques autorisent ou refusent cette action pour cette ressource. Vous pouvez également fournir des valeurs pour toutes les clés de contexte auxquelles vos politiques font référence. Vous pouvez obtenir la liste des clés de contexte auxquelles les politiques font référence en appelant d'abord `GetContextKeysForCustomPolicy` ou `GetContextKeysForPrincipalPolicy`. Si vous ne fournissez pas de valeur pour une clé de contexte, la simulation s'exécute malgré tout. En revanche, les résultats peuvent ne pas être fiables, car le simulateur de politiques ne peut pas inclure cette clé de contexte dans l'évaluation.

Pour obtenir la liste des clés de contexte (AWS CLI, AWS API)

Utilisez les clés suivantes afin d'évaluer une liste de politiques et de renvoyer une liste de clés de contexte utilisées dans les politiques.

- AWS CLI : `aws iam get-context-keys-for-custom-policy` et `aws iam get-context-keys-for-principal-policy`
- AWS API : `GetContextKeysForCustomPolicy`et `GetContextKeysForPrincipalPolicy`

Pour simuler des politiques IAM (AWS CLI, AWS API)

Utilisez ce qui suit afin de simuler des politiques IAM pour déterminer les autorisations effectives d'un utilisateur.

- AWS CLI : `aws iam simulate-custom-policy` et `aws iam simulate-principal-policy`
- AWS API : `SimulateCustomPolicy`et `SimulatePrincipalPolicy`

Ajout et suppression d'autorisations basées sur l'identité IAM

Vous utilisez des politiques pour définir les autorisations pour une identité (utilisateur, groupe d'utilisateurs ou rôle). Vous pouvez ajouter et supprimer des autorisations en attachant et en détachant des politiques IAM pour une identité à l'aide de l'API AWS Management Console, du AWS Command Line Interface (AWS CLI) ou de l' AWS API. Vous pouvez aussi utiliser des politiques afin de définir les limites d'autorisations uniquement pour les entités (utilisateurs ou rôles) qui utilisent les mêmes méthodes. Les limites d'autorisations sont une AWS fonctionnalité avancée qui contrôle le maximum d'autorisations qu'une entité peut avoir.

Rubriques

- Terminologie
- Afficher l'activité d'identité
- Ajout des autorisations d'identité IAM (console)
- Suppression des autorisations d'identité IAM (console)
- Ajout de politiques IAM (AWS CLI)
- Supprimer des politiques IAM (AWS CLI)
- Ajouter des politiques IAM (AWS API)
- Supprimer les politiques IAM (AWS API)

Terminologie

Lorsque vous associez des politiques d'autorisations à des identités (utilisateurs, groupes d'utilisateurs et rôles), la terminologie et les procédures varient selon que vous utilisez ou pas une politique gérée ou en ligne :

- Attach (Attacher) : utilisé avec les politiques gérées. Vous attachez une politique gérée à une identité (utilisateur, groupe d'utilisateurs ou rôle). L'attachement d'une politique applique les autorisations de cette dernière à l'identité.
- Detach (Détacher) : utilisé avec les politiques gérées. Vous détachez une politique gérée d'une identité IAM (utilisateur, groupe d'utilisateurs ou rôle). Le détachement d'une politique entraîne la suppression des ses autorisations de l'identité.
- Embed (Intégrer) : utilisé avec les politiques en ligne. Vous intégrez une politique en ligne à une identité (utilisateur, groupe d'utilisateurs ou rôle). L'intégration d'une politique applique les autorisations de cette dernière à l'identité. Dans la mesure où une politique en ligne est stockée dans l'identité, elle est intégrée plutôt qu'attachée, même si les résultats sont similaires.

  > **Note**
  >
  > Vous ne pouvez intégrer une politique en ligne pour un rôle lié à un service que dans le service qui dépend du rôle. Veuillez consulter la documentation AWS de votre service pour savoir si celui-ci prend en charge cette fonction.

- Delete (Supprimer) : utilisé avec les politiques en ligne. Vous supprimez une politique en ligne d'une identité IAM (utilisateur, groupe d'utilisateurs ou rôle). La suppression d'une politique entraîne la suppression de ses autorisations de l'identité.

  > **Note**
  >
  > Vous ne pouvez supprimer une politique en ligne pour un rôle lié à un service que dans le service qui dépend du rôle. Veuillez consulter la documentation AWS de votre service pour savoir si celui-ci prend en charge cette fonction.

Vous pouvez utiliser la console ou AWS CLI l' AWS API pour effectuer l'une de ces actions.

En savoir plus

- Pour en savoir plus sur la différence entre les stratégies gérées et les stratégies en ligne, consultez Politiques gérées et politiques en ligne.
- Pour plus d'informations sur les limites d'autorisations, consultez Limites d'autorisations pour les entités IAM.
- Pour obtenir des informations d'ordre général sur les politiques IAM, consultez Politiques et autorisations dans IAM.
- Pour plus d'informations sur la validation des politiques IAM, veuillez consulter Validation de politiques IAM.
- Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d'informations, consultez IAM et quotas AWS STS.

Afficher l'activité d'identité

Avant de modifier les autorisations d'une identité (utilisateur, groupe d'utilisateurs ou rôle), vous devez examiner leur activité récente au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Ajout des autorisations d'identité IAM (console)

Vous pouvez utiliser le AWS Management Console pour ajouter des autorisations à une identité (utilisateur, groupe d'utilisateurs ou rôle). Pour ce faire, attachez les stratégies gérées qui contrôlent les autorisations, ou spécifiez une stratégie qui sert de limite d'autorisations. Vous pouvez également intégrer une politique en ligne.

Pour utiliser une politique gérée en tant que politique d'autorisations pour une identité (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des politiques, sélectionnez la case d'option en regard du nom de la politique à attacher. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Sélectionnez Actions, puis Attach (Attacher).

5. Sélectionnez une ou plusieurs identités auxquelles attacher la politique. Vous pouvez utiliser la zone de recherche pour filtrer la liste des entités principales. Après avoir sélectionné les identités, choisissez Attacher une politique.

Pour utiliser une politique gérée permettant de définir une limite d'autorisations (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de celle à modifier. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Sur la page des détails de la politique, choisissez l'onglet Entités attachées, puis, si nécessaire, ouvrez la section Attachées en tant que limites des autorisations et choisissez Définir cette politique en tant que limite des autorisations.
5. Sélectionnez un ou plusieurs utilisateurs ou rôles sur lesquels utiliser la politique comme limite d'autorisations. Vous pouvez utiliser la zone de recherche pour filtrer la liste des entités principales. Après avoir sélectionné les principaux, choisissez Définir la limite des autorisations.

Pour intégrer une politique en ligne pour un utilisateur ou un rôle (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Users (Utilisateurs) ou Roles (Rôles).
3. Dans la liste, sélectionnez le nom de l'utilisateur ou du rôle auquel intégrer une politique.
4. Choisissez l’onglet Permissions (Autorisations).
5. Sélectionnez Ajouter des autorisations, puis Ajouter la politique.

> **Note**
>
> Vous ne pouvez pas intégrer une politique en ligne dans un rôle lié à un service dans IAM. Comme le service lié définit si vous pouvez modifier les autorisations du rôle, vous pourrez peut-être ajouter des politiques depuis la console de service, l'API ou l’interface AWS CLI. Pour afficher la documentation d'un rôle lié à un service, consultez

AWS services qui fonctionnent avec IAM et choisissez Oui dans la colonne Rôle lié à un service de votre service.

6. Faites votre choix parmi les méthodes suivantes pour afficher les étapes requises pour créer votre politique :

   - Importation de politiques gérées existantes : vous pouvez importer une politique gérée dans votre compte, puis la modifier afin de la personnaliser en fonction des vos exigences spécifiques. Une politique gérée peut être une politique AWS gérée ou une politique gérée par le client que vous avez créée précédemment.
   - Création de politiques avec l'éditeur visuel : vous pouvez construire intégralement une nouvelle politique dans l'éditeur visuel. Si vous utilisez l'éditeur visuel, vous n'avez pas besoin de comprendre la syntaxe JSON.
   - Création de politiques à l'aide de l'éditeur JSON : dans l'option éditeur JSON, vous pouvez créer une politique à l'aide de la syntaxe JSON. Vous pouvez composer un nouveau document de stratégie JSON ou coller un exemple de stratégie.

7. Une fois que vous avez créé une politique en, ligne, elle est automatiquement intégrée à votre utilisateur ou rôle.

Pour intégrer une politique en ligne pour un groupe d'utilisateurs (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs).
3. Dans la liste, sélectionnez le nom du groupe d'utilisateurs auquel intégrer une politique.
4. Choisissez l'onglet Permissions (Autorisations), puis Add permissions (Ajouter des autorisations), et enfin Create inline policy (Créer une politique en ligne).
5. Effectuez l’une des actions suivantes :

   - Sélectionnez l'option Visuel pour créer la politique. Pour plus d’informations, consultez Création de politiques avec l'éditeur visuel.
   - Sélectionnez l'option JSON pour créer la politique. Pour plus d’informations, consultez Création de politiques à l'aide de l'éditeur JSON.

6. Lorsque vous êtes satisfait de la politique, choisissez Créer une politique.

Pour modifier la limite d'autorisations d'une ou plusieurs entités (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de celle à modifier. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Sur la page des détails de la politique, choisissez l'onglet Entités attachées, puis, si nécessaire, ouvrez la section Attachées en tant que limite des autorisations. Cochez la case en regard des utilisateurs ou des rôles dont vous voulez modifier les limites, puis choisissez Modifier.
5. Sélectionnez une nouvelle politique à utiliser comme limite d'autorisations. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques. Après avoir sélectionné la politique, choisissez Définir la limite des autorisations.

Suppression des autorisations d'identité IAM (console)

Vous pouvez utiliser le AWS Management Console pour supprimer les autorisations associées à une identité (utilisateur, groupe d'utilisateurs ou rôle). Pour ce faire, détachez les politiques gérées qui contrôlent les autorisations, ou supprimez une politique qui a servi de limite d'autorisations. Vous pouvez également supprimer une politique en ligne.

Pour détacher une politique gérée utilisée en tant que politique d'autorisations (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des politiques, sélectionnez la case d'option en regard du nom de la politique à détacher. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Sélectionnez Actions, puis Detach (Détacher).
5. Sélectionnez les identités desquelles détacher la politique. Vous pouvez utiliser la zone de recherche pour filtrer la liste des identités. Après avoir sélectionné les identités, choisissez Détacher la politique.

Pour supprimer une limite d'autorisations (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de celle à modifier. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Sur la page de résumé de la politique, choisissez l'onglet Entités attachées, puis, si nécessaire, ouvrez la section Attachée en tant que limite des autorisations et choisissez les entités pour lesquelles vous supprimez la limite des autorisations. Puis, choisissez Supprimer la limite.
5. Confirmez la suppression de la limite et choisissez Supprimer la limite.

Pour supprimer une politique en ligne (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), Users (Utilisateurs) ou Roles (Rôles).
3. Choisissez dans la liste le nom du groupe d'utilisateurs, de l'utilisateur ou du rôle pour lequel vous souhaitez supprimer la politique.
4. Choisissez l’onglet Permissions (Autorisations).
5. Cochez la case en regard de la politique et choisissez Supprimer.
6. Dans la boîte de confirmation, choisissez Supprimer.

Ajout de politiques IAM (AWS CLI)

Vous pouvez utiliser le AWS CLI pour ajouter des autorisations à une identité (utilisateur, groupe d'utilisateurs ou rôle). Pour ce faire, attachez les stratégies gérées qui contrôlent les autorisations, ou spécifiez une stratégie qui sert de limite d'autorisations. Vous pouvez également intégrer une politique en ligne.

Pour utiliser une politique gérée en tant que politique d'autorisations pour une entité (AWS CLI)

1. (Facultatif) Pour afficher les informations d'une politique gérée, exécutez les commandes suivantes :

- Pour répertorier les stratégies gérées : aws iam list-policies
- Pour récupérer les informations détaillées sur une politique gérée : get-policy

2. Pour attacher une politique gérée à une identité (utilisateur, groupe d'utilisateurs ou rôle), utilisez l'une des commandes suivantes :

   - était un objectif attach-user-policy
   - était un objectif attach-group-policy
   - était un objectif attach-role-policy

Pour utiliser une politique gérée permettant de définir une limite d'autorisations (AWS CLI)

1. (Facultatif) Pour afficher les informations d'une politique gérée, exécutez les commandes suivantes :

   - Pour répertorier les stratégies gérées : aws iam list-policies
   - Pour récupérer les informations détaillées sur une politique gérée : aws iam get-policy

2. Pour utiliser une politique gérée permettant de définir la limite d'autorisations pour une entité (utilisateur ou rôle), utilisez l'une des commandes suivantes :

   - était un objectif put-user-permissions-boundary
   - était un objectif put-role-permissions-boundary

Pour intégrer une politique en ligne (AWS CLI)

Pour intégrer une politique en ligne à une identité (utilisateur, groupe d'utilisateurs ou rôle qui n'est pas un rôle lié à un service), utilisez l'une des commandes suivantes :

- était un objectif put-user-policy
- était un objectif put-group-policy
- était un objectif put-role-policy

Supprimer des politiques IAM (AWS CLI)

Vous pouvez utiliser le AWS CLI pour détacher les politiques gérées qui contrôlent les autorisations ou supprimer une politique qui sert de limite d'autorisations. Vous pouvez également supprimer une politique en ligne.

Pour détacher une politique gérée utilisée en tant que politique d'autorisations (AWS CLI)

1. (Facultatif) Pour afficher des informations sur une politique, exécutez les commandes suivantes :
   - Pour répertorier les stratégies gérées : aws iam list-policies
   - Pour récupérer les informations détaillées sur une politique gérée : aws iam get-policy
2. (Facultatif) Pour en savoir plus sur les relations entre les stratégies et les identités, exécutez les commandes suivantes :
   - Pour répertorier les identités (utilisateurs, groupes d'utilisateurs et rôles) auxquelles une politique gérée est attachée :
     - était un objectif list-entities-for-policy
   - Pour répertorier les politiques gérées attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle), utilisez l'une des commandes suivantes :
     - était un objectif list-attached-user-policies
     - était un objectif list-attached-group-policies
     - était un objectif list-attached-role-policies
3. Pour détacher une politique gérée d'une identité (utilisateur, groupe d'utilisateurs ou rôle), utilisez l'une des commandes suivantes :
   - était un objectif detach-user-policy
   - était un objectif detach-group-policy
   - était un objectif detach-role-policy

Pour supprimer une limite d'autorisations (AWS CLI)

1. (Facultatif)Pour afficher la politique gérée actuellement utilisée pour définir la limite d'autorisations d'un utilisateur ou rôle, exécutez les commandes suivantes :
   - aws iam get-user
   - aws iam get-role

2. (Facultatif)Pour afficher les utilisateurs ou rôles sur lesquels une politique gérée est utilisée pour une limite d'autorisations, exécutez la commande suivante :

   - était un objectif list-entities-for-policy

3. (Facultatif) Pour afficher les informations d'une politique gérée, exécutez les commandes suivantes :

   - Pour répertorier les stratégies gérées : aws iam list-policies
   - Pour récupérer les informations détaillées sur une politique gérée : aws iam get-policy

4. Pour supprimer une limite d'autorisations d'un utilisateur ou d'un rôle, utilisez l'une des commandes suivantes :

   - était un objectif delete-user-permissions-boundary
   - était un objectif delete-role-permissions-boundary

Pour supprimer une politique en ligne (AWS CLI)

1. (Facultatif) Pour répertorier toutes les politiques en ligne attachées à une identité (utilisateur, groupe d'utilisateurs, rôle), utilisez l'une des commandes suivantes :

   - était un objectif list-user-policies
   - était un objectif list-group-policies
   - était un objectif list-role-policies

2. (Facultatif) Pour récupérer un document de politique en ligne qui est intégré à une identité (utilisateur, groupe d'utilisateurs ou rôle), utilisez l'une des commandes suivantes :

   - était un objectif get-user-policy
   - était un objectif get-group-policy
   - était un objectif get-role-policy

3. Pour supprimer une politique en ligne d'une identité (utilisateur, groupe d'utilisateurs ou rôle qui n'est pas un rôle lié à un service), utilisez l'une des commandes suivantes :

   - était un objectif delete-user-policy
   - était un objectif delete-group-policy
   - était un objectif delete-role-policy

Ajouter des politiques IAM (AWS API)

Vous pouvez utiliser l' AWS API pour associer des politiques gérées qui contrôlent les autorisations ou spécifier une politique qui sert de limite d'autorisations. Vous pouvez également intégrer une politique en ligne.

Pour utiliser une politique gérée comme politique d'autorisation pour une entité (AWS API)

1. (Facultatif) Pour afficher des informations sur une politique, appelez les opérations suivantes :
   - Pour répertorier les politiques gérées : ListPolicies
   - Pour récupérer des informations détaillées sur une politique gérée, procédez comme suit : GetPolicy
2. Pour attacher une politique gérée à une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :
   - AttachUserPolicy
   - AttachGroupPolicy
   - AttachRolePolicy

Pour utiliser une politique gérée afin de définir une limite d'autorisations (AWS API)

1. (Facultatif) Pour afficher les informations d'une politique gérée, appelez les opérations suivantes :
   - Pour répertorier les politiques gérées : ListPolicies
   - Pour récupérer des informations détaillées sur une politique gérée, procédez comme suit : GetPolicy
2. Pour utiliser une politique gérée permettant de définir la limite d'autorisations pour une entité (utilisateur ou rôle), appelez l'une des opérations suivantes :
   - PutUserPermissionsBoundary
   - PutRolePermissionsBoundary

Pour intégrer une politique en ligne (API)AWS

Pour intégrer une politique en ligne à une identité (utilisateur, groupe d'utilisateurs ou rôle qui n'est pas un rôle lié à un service), appelez l'une des opérations suivantes :

- PutUserPolicy
- PutGroupPolicy
- PutRolePolicy

Supprimer les politiques IAM (AWS API)

Vous pouvez utiliser l' AWS API pour détacher les politiques gérées qui contrôlent les autorisations ou supprimer une politique qui sert de limite d'autorisations. Vous pouvez également supprimer une politique en ligne.

Pour détacher une politique gérée utilisée comme politique d'autorisation (AWS API)

1. (Facultatif) Pour afficher des informations sur une politique, appelez les opérations suivantes :
   - Pour répertorier les politiques gérées : ListPolicies
   - Pour récupérer des informations détaillées sur une politique gérée, procédez comme suit : GetPolicy
2. (Facultatif) Pour en savoir plus sur les relations entre les stratégies et les identités, appelez les opérations suivantes :
   - Pour répertorier les identités (utilisateurs, groupes d'utilisateurs et rôles) auxquelles une politique gérée est attachée :
     - ListEntitiesForPolicy
   - Pour répertorier les politiques gérées attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies
3. Pour détacher une politique gérée d'une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :
   - DetachUserPolicy
   - DetachGroupPolicy
   - DetachRolePolicy

Pour supprimer une limite d'autorisation (AWS API)

1. (Facultatif)Pour afficher la politique gérée actuellement utilisée pour définir la limite d'autorisations d'un utilisateur ou rôle, appelez les opérations suivantes :
   - GetUser
   - GetRole
2. (Facultatif)Pour afficher les utilisateurs ou rôles sur lesquels une politique gérée est utilisée pour une limite d'autorisations, appelez l'opération suivante :
   - ListEntitiesForPolicy
3. (Facultatif) Pour afficher les informations d'une politique gérée, appelez les opérations suivantes :
   - Pour répertorier les politiques gérées : ListPolicies
   - Pour récupérer des informations détaillées sur une politique gérée, procédez comme suit : GetPolicy
4. Pour supprimer une limite d'autorisations d'un utilisateur ou d'un rôle, appelez l'une des opérations suivantes :
   - DeleteUserPermissionsBoundary
   - DeleteRolePermissionsBoundary

Pour supprimer une politique intégrée (AWS API)

1. (Facultatif) Pour répertorier toutes les politiques en ligne attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :
   - ListUserPolicies
   - ListGroupPolicies
   - ListRolePolicies
2. (Facultatif) Pour récupérer un document de politique en ligne qui est intégré à une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :
   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy

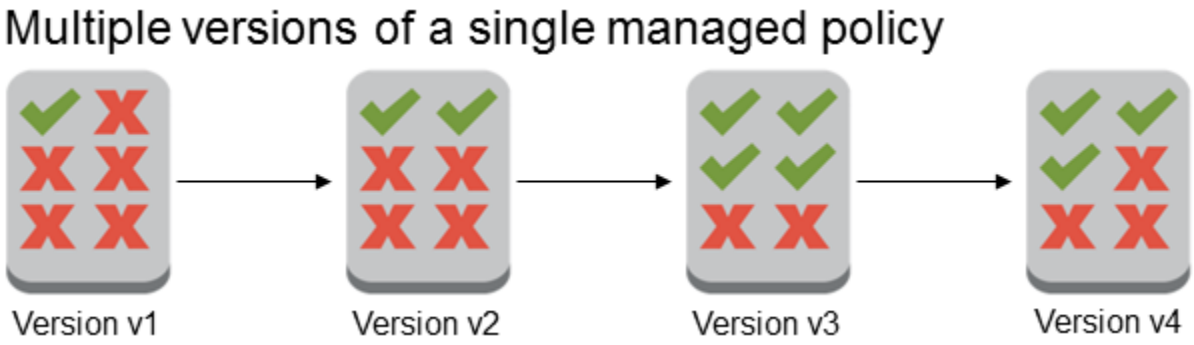
3. Pour supprimer une politique en ligne d'une identité (utilisateur, groupe d'utilisateurs ou rôle qui n'est pas un rôle lié à un service), appelez l'une des opérations suivantes :

   - DeleteUserPolicy
   - DeleteGroupPolicy
   - DeleteRolePolicy

Gestion des versions des politiques IAM

Lorsque vous apportez des modifications à une politique gérée par un client IAM ou à une politique AWS gérée, la stratégie modifiée ne remplace pas la politique existante. AWS À la place, IAM crée une nouvelle version de la politique gérée. IAM stocke jusqu'à cinq versions de vos politiques gérées par le client. IAM ne prend pas en charge la gestion des versions pour les politiques en ligne.

Le diagramme suivant illustre la gestion des versions pour une politique gérée par le client. Dans cet exemple, les versions de 1 à 4 sont enregistrées. Vous pouvez enregistrer jusqu'à cinq versions de politique gérées dans IAM. Lorsque vous créez une sixième version en modifiant une politique, vous pouvez choisir quelle version plus ancienne supprimer. Vous pouvez revenir à tout moment à l'une des quatre autres versions enregistrées.



Une version de politique est différente d'un élément de politique `Version`. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. Pour en savoir plus sur l'élément de politique `Version`, consultez Éléments de politique JSON IAM : Version.

Vous pouvez utiliser ces versions pour effectuer le suivi d'une politique gérée. Par exemple, vous pouvez modifier une politique gérée et découvrir ensuite que le changement a eu des effets inattendus. Dans ce cas, vous pouvez revenir à un version précédente de la politique gérée en définissant la version précédente comme version par défaut.

Les rubriques suivantes expliquent comment utiliser la gestion des versions avec les politiques gérées.

Rubriques

- Autorisations pour définir la version par défaut d'une politique
- Définition de la version par défaut des politiques gérées par le client
- Utilisation de versions pour annuler des modifications
- Limites de version

Autorisations pour définir la version par défaut d'une politique

Les autorisations qui sont requises pour définir la version par défaut d'une politique correspondent aux opérations d'API AWS pour la tâche. Vous pouvez utiliser les opérations d'API `CreatePolicyVersion` ou `SetDefaultPolicyVersion` pour définir la version par défaut d'une politique. Pour autoriser une personne à définir la version par défaut d'une politique existante, vous pouvez autoriser l'accès à l'action `iam:CreatePolicyVersion` ou l'action `iam:SetDefaultPolicyVersion`. L'action `iam:CreatePolicyVersion` permet de créer une nouvelle version de la politique et de définir cette version comme version par défaut. L'action `iam:SetDefaultPolicyVersion` permet de définir n'importe quelle version existante de la politique comme valeur par défaut.

> **Important**
>
> Refuser l'action `iam:SetDefaultPolicyVersion` dans une politique d'utilisateur n'empêche pas l'utilisateur de créer une nouvelle version de la politique et de la définir comme version par défaut.

Vous pouvez utiliser la politique suivante pour interdire à un utilisateur de modifier une politique existante gérée par le client :

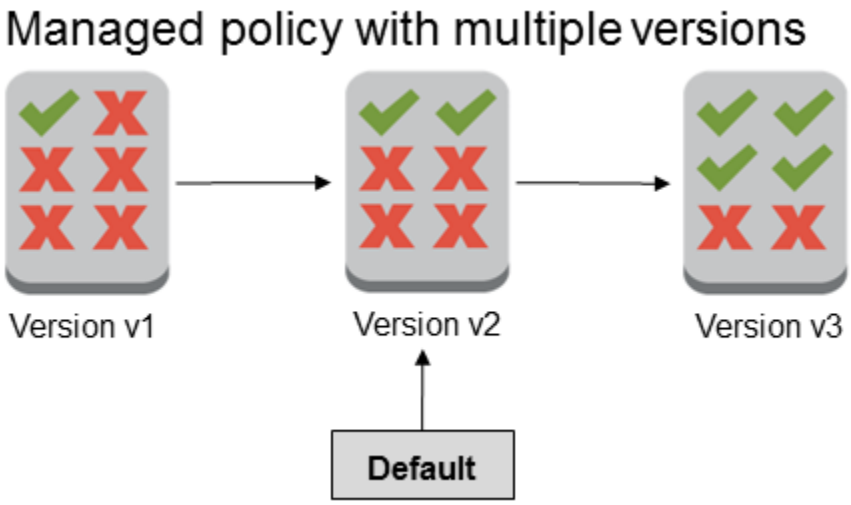
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
```

```
                "iam:CreatePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": "arn:aws:iam::*:policy/POLICY-NAME"
        }
    ]
}
```

Définition de la version par défaut des politiques gérées par le client

L'une des versions d'une politique gérée est définie comme la version par défaut. La version par défaut de la politique est la version d'exécution, c'est-à-dire qu'il s'agit de la version en vigueur pour toutes les entités principaux (utilisateurs, groupes d'utilisateurs et rôles) auxquelles la politique gérée est attachée.

Lorsque vous créez une politique gérée par le client, la politique commence par une version unique identifiée comme v1. Pour les politiques gérées avec une version unique, cette version est définie automatiquement comme valeur par défaut. Pour les politiques gérées par le client avec plusieurs versions, vous choisissez la version à définir par défaut. Pour les politiques AWS gérées, la version par défaut est définie par AWS. Les diagrammes suivants illustrent ce concept.



Vous pouvez définir la version par défaut d'une politique gérée par le client pour appliquer cette version à chaque entité IAM (utilisateur, groupe d'utilisateurs et rôle) à laquelle la politique gérée est attachée. Vous ne pouvez pas définir la version par défaut d'une politique AWS gérée ou d'une politique intégrée.

Pour définir la version par défaut d'une politique gérée par le client (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de la stratégie à définir comme la version par défaut. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Choisissez l'onglet Versions de politique. Cochez la case située en regard de la version que vous souhaitez définir comme version par défaut, puis choisissez Définir par défaut.

Pour savoir comment définir la version par défaut d'une politique gérée par le client à partir de l'API AWS Command Line Interface ou de l' AWS API, consultezModification des politiques gérées par le client (AWS CLI).

Utilisation de versions pour annuler des modifications

Vous pouvez définir la version par défaut d'une politique gérée par le client pour annuler vos modifications. Par exemple, imaginez le scénario suivant:

Vous créez une politique gérée par le client qui autorise les utilisateurs à administrer un compartiment Amazon S3 spécifique à l'aide de la AWS Management Console. À sa création, votre politique gérée par le client dispose d'une version unique, identifiée comme v1. Cette version est donc définie automatiquement comme valeur par défaut. La politique fonctionne comme prévu.

Plus tard, vous mettrez à jour la politique pour ajouter l'autorisation d'administrer un second compartiment Amazon S3. IAM crée une nouvelle version de la politique, identifiée en tant que v2, qui contient vos modifications. Vous définissez la version v2 comme version par défaut, et peu de temps après, vos utilisateurs signalent qu'ils n'ont pas l'autorisation d'utiliser la console Amazon S3. Dans ce cas, vous pouvez revenir à la version v1 de la politique, que vous savez fonctionner comme prévu. Pour cela, vous définissez la version v1 comme version par défaut. Vous utilisateurs sont désormais en mesure d'utiliser la console Amazon S3 pour administrer le compartiment d'origine.

Plus tard, après avoir identifié l'erreur dans la version v2 de la politique, vous remettez à jour la politique pour ajouter l'autorisation d'administrer le second compartiment Amazon S3. IAM crée une autre version de la politique, identifiée en tant que v3. Vous définissez la version v3 comme version par défaut et cette version fonctionne comme prévu. À ce stade, vous supprimez la version v2 de la politique.

Limites de version

Une politique gérée peut avoir jusqu'à cinq versions. Si vous devez apporter des modifications à une politique gérée au-delà de cinq versions à partir de l' AWS Command Line Interface API ou de l' AWS API, vous devez d'abord supprimer une ou plusieurs versions existantes. Si vous utilisez le AWS Management Console, il n'est pas nécessaire de supprimer une version avant de modifier votre politique. Lorsque vous disposez d'une sixième version, une boîte de dialogue s'affiche pour vous inviter à supprimer une ou plusieurs versions personnalisées de votre politique. Vous pouvez également afficher le document de politique JSON pour vous aider à choisir. Pour plus d'informations sur cette boîte de dialogue, consultez la section the section called “Modification de politiques IAM”.

Vous pouvez supprimer n'importe quelle version de la politique gérée de votre choix, sauf la version par défaut. Lorsque vous supprimez une version, les identificateurs de version des versions restantes ne changent pas. En conséquence, les identificateurs de version peuvent ne pas être séquentiels. Par exemple, si vous supprimez les versions v2 et v4 d'une politique gérée et que vous ajoutez deux nouvelles versions, les identificateurs de version restants peuvent être v1, v3, v5, v6 et v7.

Modification de politiques IAM

Une politique est une entité qui, lorsqu'elle est attachée une identité ou à une ressource, définit les autorisations de cette dernière. Les politiques sont stockées AWS sous forme de documents JSON et sont attachées aux principes sous forme de politiques basées sur l'identité dans IAM. Vous pouvez attacher une politique basée sur identité à un principal (ou une identité), comme un groupe d'utilisateurs, un utilisateur ou un rôle IAM. Les politiques basées sur l'identité incluent les politiques AWS gérées, les politiques gérées par le client et les politiques intégrées. Vous pouvez modifier les politiques gérées par le client et les politiques intégrées dans IAM. AWS les politiques gérées ne peuvent pas être modifiées. Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d’informations, consultez IAM et quotas AWS STS.

Rubriques

- Afficher l'accès à la politique
- Modification de politiques gérées par le client (console)
- Modification de politiques en ligne (console)
- Modification des politiques gérées par le client (AWS CLI)
- Modification des politiques gérées par le client (AWS API)

Afficher l'accès à la politique

Avant de modifier les autorisations d'une politique, vous devez passer en revue ses activités récentes au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Modification de politiques gérées par le client (console)

Vous pouvez modifier les politiques gérées par le client pour modifier les autorisations définies dans la politique. Une politique gérée le client peut avoir jusqu'à cinq versions. Ceci est important car si vous apportez des modifications à une politique gérée au-delà de cinq versions, le AWS Management Console système vous invite à choisir la version à supprimer. Vous pouvez également modifier la version par défaut ou supprimer une version d'une politique avant de modifier celle-ci a pour éviter d'y être invité. Pour en savoir plus sur les versions, consultez Gestion des versions des politiques IAM.

Pour modifier une politique gérée par le client (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des politiques, choisissez le nom de la politique à modifier. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Choisissez l'onglet Autorisations, puis Modifier.
5. Effectuez l'une des actions suivantes :

   - Choisissez l'option Visuel pour modifier votre politique sans comprendre la syntaxe JSON. Vous pouvez apporter des modifications au service, aux ressources d'actions ou à des conditions facultatives pour chaque bloc d'autorisation de votre politique. Vous pouvez également importer une politique pour ajouter des autorisations supplémentaires au bas de votre politique. Lorsque vous avez fini d'apporter des modifications, choisissez Suivant pour continuer.
   - Choisissez l'option JSON pour modifier votre politique en tapant ou en collant du texte dans la zone de texte JSON. Vous pouvez également importer une politique pour ajouter des autorisations supplémentaires au bas de votre politique. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant.

   > **Note**
   >
   > Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez Restructuration de politique.

6. Sur la page Vérifier et enregistrer, vérifiez les Autorisations définies dans cette politique, puis choisissez Enregistrer les modifications pour enregistrer votre travail.
7. Si la politique gérée dispose déjà du maximum de cinq versions, une boîte de dialogue s'affiche lorsque vous choisissez Enregistrer les modifications. Pour enregistrer votre nouvelle version, la version la plus ancienne de la politique est supprimée et remplacée par cette nouvelle version. Vous pouvez choisir de définir la nouvelle version en tant que version par défaut de la politique.

   Choisissez Enregistrer les modifications pour enregistrer votre nouvelle version de la politique.

Pour définir la version par défaut d'une politique gérée par le client (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de la stratégie à définir comme la version par défaut. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Choisissez l'onglet Versions de politique. Cochez la case située en regard de la version que vous souhaitez définir comme version par défaut, puis choisissez Définir par défaut.

Pour supprimer une version d'une politique gérée par le client (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Politiques.
3. Choisissez le nom de la politique gérée par le client dont vous souhaitez supprimer une version. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Choisissez l'onglet Versions de politique. Activez la case à cocher en regard de la version à supprimer. Ensuite, choisissez Supprimer.
5. Confirmez que vous voulez supprimer la version, puis choisissez Supprimer.

Modification de politiques en ligne (console)

Vous pouvez modifier une politique en ligne à partir de l’interface AWS Management Console.

Pour modifier une politique en ligne pour un utilisateur, un groupe d'utilisateurs ou un rôle (console)

1. Dans le panneau de navigation, sélectionnez Users (Utilisateurs), User groups (Groupes d'utilisateurs) ou Roles (Rôles).
2. Choisissez le nom de l'utilisateur, du groupe d'utilisateurs ou du rôle ayant la politique que vous souhaitez modifier. Choisissez ensuite l'onglet Autorisations et développez la politique.
3. Pour modifier une politique en ligne, choisissez Modifier la politique.
4. Effectuez l’une des actions suivantes :

   - Choisissez l'option Visuel pour modifier votre politique sans comprendre la syntaxe JSON. Vous pouvez apporter des modifications au service, aux ressources d’actions ou à des

conditions facultatives pour chaque bloc d'autorisation de votre politique. Vous pouvez également importer une politique pour ajouter des autorisations supplémentaires au bas de votre politique. Lorsque vous avez fini d'apporter des modifications, choisissez Suivant pour continuer.

- Choisissez l'option JSON pour modifier votre politique en tapant ou en collant du texte dans la zone de texte JSON. Vous pouvez également importer une politique pour ajouter des autorisations supplémentaires au bas de votre politique. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant. Pour enregistrer vos modifications sans affecter les entités actuellement attachées, désactivez la case à cocher Définir comme version par défaut.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez Restructuration de politique.

5. Sur la page Vérifier, vérifiez le résumé de la politique, puis choisissez Enregistrer les modifications pour enregistrer votre travail.

Modification des politiques gérées par le client (AWS CLI)

Vous pouvez modifier une politique gérée par le client à partir du AWS Command Line Interface (AWS CLI).

> **Note**
>
> Une politique gérée peut avoir jusqu'à cinq versions. Si vous devez modifier une politique gérée par le client au-delà de cinq versions, vous devez d'abord supprimer une ou plusieurs versions existantes.

Pour modifier une politique gérée par le client (AWS CLI)

1. (Facultatif) Pour afficher des informations sur une politique, exécutez les commandes suivantes :

- Pour répertorier les stratégies gérées : list-policies
- Pour récupérer les informations détaillées sur une politique gérée : get-policy

2. (Facultatif) Pour en savoir plus sur les relations entre les stratégies et les identités, exécutez les commandes suivantes :

   - Pour répertorier les identités (utilisateurs, groupes d'utilisateurs et rôles) auxquelles une politique gérée est attachée :
     - list-entities-for-policy
   - Pour répertorier les politiques gérées attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle) :
     - list-attached-user-policies
     - list-attached-group-policies
     - list-attached-role-policies

3. Pour modifier une politique gérée par le client, exécutez la commande suivante :

   - create-policy-version

4. (Facultatif) Pour valider une politique gérée par le client, exécutez la commande IAM Access Analyzer suivante :

   - validate-policy

Pour définir la version par défaut d'une politique gérée par le client (AWS CLI)

1. (Facultatif) Pour répertorier les stratégies gérées, exécutez la commande suivante :

   - list-policies

2. Pour définir la version par défaut d'une politique gérée par le client, exécutez la commande suivante :

   - set-default-policy-version

Pour supprimer une version d'une politique gérée par le client (AWS CLI)

1. (Facultatif) Pour répertorier les stratégies gérées, exécutez la commande suivante :

   - list-policies

2. Pour supprimer une politique gérée par le client, exécutez la commande suivante :

   - delete-policy-version

Modification des politiques gérées par le client (AWS API)

Vous pouvez modifier une politique gérée par le client à l'aide de l' AWS API.

> **Note**
>
> Une politique gérée peut avoir jusqu'à cinq versions. Si vous devez modifier une politique gérée par le client au-delà de cinq versions, vous devez d'abord supprimer une ou plusieurs versions existantes.

Pour modifier une politique gérée par le client (AWS API)

1. (Facultatif) Pour afficher des informations sur une politique, appelez les opérations suivantes :

   - Pour répertorier les politiques gérées : ListPolicies
   - Pour récupérer des informations détaillées sur une politique gérée, procédez comme suit : GetPolicy

2. (Facultatif) Pour en savoir plus sur les relations entre les stratégies et les identités, appelez les opérations suivantes :

   - Pour répertorier les identités (utilisateurs, groupes d'utilisateurs et rôles) auxquelles une politique gérée est attachée :
     - ListEntitiesForPolicy
   - Pour répertorier les politiques gérées attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle) :
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies

3. Pour modifier une politique gérée par le client, appelez l'opération suivante :

   - CreatePolicyVersion

4. (Facultatif) Pour valider une politique gérée par le client, appelez l'opération IAM Access Analyzer suivante :

   - ValidatePolicy

Pour définir la version par défaut d'une politique gérée par le client (AWS API)

1. (Facultatif) Pour répertorier les stratégies gérées, appelez l'opération suivante :

   - ListPolicies
2. Pour définir la version par défaut d'une politique gérée par le client, appelez l'opération suivante :

   - SetDefaultPolicyVersion

Pour supprimer une version d'une politique gérée par le client (AWS API)

1. (Facultatif) Pour répertorier les stratégies gérées, appelez l'opération suivante :

   - ListPolicies
2. Pour supprimer une politique gérée par le client, appelez l'opération suivante :

   - DeletePolicyVersion

Suppression de politiques IAM

Vous pouvez supprimer les politiques IAM à l'aide de l' AWS Management Console API, de la AWS Command Line Interface (AWS CLI) ou de l'API IAM.

> **Note**
>
> Toute suppression d'une politique IAM est définitive. Une fois la politique supprimée, elle ne peut plus être récupérée.

Pour en savoir plus sur la différence entre les stratégies gérées et les stratégies en ligne, consultez Politiques gérées et politiques en ligne.

Pour obtenir des informations d'ordre général sur les politiques IAM, consultez Politiques et autorisations dans IAM.

Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d'informations, consultez IAM et quotas AWS STS.

Rubriques

- Afficher l'accès à la politique
- Création de politiques de rôle IAM (console)
- Suppression de politiques IAM AWS CLI
- Supprimer des politiques IAM (AWS API)

Afficher l'accès à la politique

Avant de supprimer une politique, vous devez passer en revue ses activités récentes au niveau service. Ceci est important, car vous ne souhaitez pas supprimer l'accès à partir d'un principal (personne ou application) qui l'utilise. Pour de plus amples informations sur l'affichage des dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Création de politiques de rôle IAM (console)

Vous pouvez supprimer une politique gérée par le client pour la retirer de votre Compte AWS. Vous ne pouvez pas supprimer les politiques AWS gérées.

Pour supprimer une politique gérée par le client (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Politiques.
3. Sélectionnez la case d'option en regard de la politique gérée par le client à supprimer. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Choisissez Actions, puis choisissez Supprimer.
5. Suivez les instructions pour confirmer la suppression de la politique, puis choisissez Supprimer.

Pour supprimer une politique en ligne pour un groupe d'utilisateurs, un utilisateur ou un rôle (console)

1. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), Users (Utilisateurs) ou Roles (Rôles).

2. Choisissez le nom du groupe d'utilisateurs, de l'utilisateur ou du rôle et la politique que vous souhaitez supprimer. Ensuite, choisissez l'onglet Autorisations.
3. Cochez les cases en regard des politiques à supprimer et choisissez Supprimer. Pour supprimer une politique en ligne dans Utilisateurs ou Rôles, choisissez Supprimer pour confirmer la suppression. Si vous supprimez une seule politique en ligne dansGroupes d'utilisateurs, tapez le nom de la politique et choisissezSupprimer. Si vous supprimez plusieurs politiques en ligne dans User groups (Groupes d'utilisateurs), tapez le nombre de politiques à supprimer, suivi de **`inline policies`**, puis choisissez Delete (Supprimer). Par exemple, si vous supprimez trois politiques en ligne, tapez **`3 inline policies`**.

Suppression de politiques IAM AWS CLI

Vous pouvez supprimer une politique gérée par le client à partir de l'interface AWS Command Line Interface.

Pour supprimer une politique gérée par le client (AWS CLI)

1. (Facultatif) Pour afficher des informations sur une politique, exécutez les commandes suivantes :
   - Pour répertorier les stratégies gérées : list-policies
   - Pour récupérer les informations détaillées sur une politique gérée : get-policy
2. (Facultatif) Pour en savoir plus sur les relations entre les stratégies et les identités, exécutez les commandes suivantes :
   - Pour répertorier les identités (utilisateurs, groupes d'utilisateurs et rôles) auxquelles une politique gérée est attachée, exécutez la commande suivante :
     - list-entities-for-policy
   - Pour répertorier les politiques gérées attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle), exécutez l'une des commandes suivantes :
     - list-attached-user-policies
     - list-attached-group-policies
     - list-attached-role-policies
3. Pour supprimer une politique gérée par le client, exécutez la commande suivante :
   - delete-policy

Pour supprimer une politique en ligne (AWS CLI)

1. (Facultatif) Pour répertorier toutes les politiques en ligne attachées à une identité (utilisateur, groupe d'utilisateurs, rôle), utilisez l'une des commandes suivantes :
   - était un objectif list-user-policies
   - était un objectif list-group-policies
   - était un objectif list-role-policies
2. (Facultatif) Pour récupérer un document de politique en ligne qui est intégré à une identité (utilisateur, groupe d'utilisateurs ou rôle), utilisez l'une des commandes suivantes :
   - était un objectif get-user-policy
   - était un objectif get-group-policy
   - était un objectif get-role-policy
3. Pour supprimer une politique en ligne d'une identité (utilisateur, groupe d'utilisateurs ou rôle qui n'est pas un rôle lié à un service), utilisez l'une des commandes suivantes :
   - était un objectif delete-user-policy
   - était un objectif delete-group-policy
   - était un objectif delete-role-policy

Supprimer des politiques IAM (AWS API)

Vous pouvez supprimer une politique gérée par le client à l'aide de l' AWS API.

Pour supprimer une politique gérée par le client (AWS API)

1. (Facultatif) Pour afficher des informations sur une politique, appelez les opérations suivantes :
   - Pour répertorier les politiques gérées : ListPolicies
   - Pour récupérer des informations détaillées sur une politique gérée, procédez comme suit : GetPolicy
2. (Facultatif) Pour en savoir plus sur les relations entre les stratégies et les identités, appelez les opérations suivantes :
   - Pour répertorier les identités (utilisateurs, groupes d'utilisateurs et rôles) auxquelles une politique gérée est attachée, appelez l'opération suivante :

- ListEntitiesForPolicy
- Pour répertorier les politiques gérées attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :
  - ListAttachedUserPolicies
  - ListAttachedGroupPolicies
  - ListAttachedRolePolicies

3. Pour supprimer une politique gérée par le client, appelez l'opération suivante :
   - DeletePolicy

Pour supprimer une politique intégrée (AWS API)

1. (Facultatif) Pour répertorier toutes les politiques en ligne attachées à une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :
   - ListUserPolicies
   - ListGroupPolicies
   - ListRolePolicies
2. (Facultatif) Pour récupérer un document de politique en ligne qui est intégré à une identité (utilisateur, groupe d'utilisateurs ou rôle), appelez l'une des opérations suivantes :
   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy
3. Pour supprimer une politique en ligne d'une identité (utilisateur, groupe d'utilisateurs ou rôle qui n'est pas un rôle lié à un service), appelez l'une des opérations suivantes :
   - DeleteUserPolicy
   - DeleteGroupPolicy
   - DeleteRolePolicy

Affiner les autorisations en AWS utilisant les dernières informations consultées

En tant qu'administrateur, vous pouvez accorder davantage d'autorisations aux ressources IAM (rôles, utilisateurs, groupes d'utilisateurs ou politiques) qu'elles n'en nécessitent. IAM fournit les dernières informations consultées pour vous aider à identifier les autorisations inutilisées et les supprimer. Vous pouvez utiliser les dernières informations consultées pour affiner vos politiques et n'autoriser l'accès qu'aux services et actions utilisés par vos identités et politiques IAM. Cela vous permet de mieux respecter la bonne pratique que l'on appelle principe du moindre privilège. Vous pouvez afficher les dernières informations consultées pour les identités ou les politiques qui existent dans IAM ou AWS Organizations.

Vous pouvez surveiller en permanence les dernières informations consultées à l'aide d'analyseurs d'accès non utilisés. Pour plus d'informations, consultez Résultats des accès externes et non utilisés.

Rubriques

- Types des dernières informations consultées pour IAM
- Dernières informations consultées pour AWS Organizations
- Choses à savoir sur les dernières informations consultées
- Autorisations nécessaires
- Activité de dépannage pour des entités IAM et Organizations
- Où AWS suit les dernières informations consultées
- Affichage des dernières informations consultées pour IAM
- Affichage des dernières informations consultées pour Organizations
- Exemples de scénarios d'utilisation des dernières informations consultées
- Services et actions concernant les dernières informations consultées relatives à une action IAM

Types des dernières informations consultées pour IAM

Vous pouvez afficher deux types de dernières informations consultées pour les identités IAM : les informations sur les services AWS autorisés et les informations sur les actions autorisées. Les informations incluent la date et l'heure auxquelles la tentative d'accès à une AWS API a été faite. Pour les actions, les dernières informations consultées font état des actions de gestion des services. Les actions de gestion comprennent les actions de création, de suppression et de modification.

Pour en savoir plus sur la façon d'afficher les dernières informations consultées pour IAM, consultez Affichage des dernières informations consultées pour IAM.

Pour obtenir des exemples de scénarios concernant l'utilisation des dernières informations consultées dans le but de prendre des décisions sur les autorisations que vous accordez à vos identités IAM, veuillez consulter Exemples de scénarios d'utilisation des dernières informations consultées.

Pour en savoir plus sur la façon dont les informations relatives aux actions de gestion sont fournies, consultez Choses à savoir sur les dernières informations consultées.

Dernières informations consultées pour AWS Organizations

Si vous vous connectez à l'aide des informations d'identification du compte de gestion, vous pouvez consulter les informations du dernier accès au service pour une AWS Organizations entité ou une politique de votre organisation. AWS Organizations les entités incluent la racine de l'organisation, les unités organisationnelles (UO) ou les comptes. Les dernières informations consultées pour AWS Organizations incluent des informations sur les services autorisés par une politique de contrôle des services (SCP). Les informations indiquent quels principaux (utilisateur root, utilisateur IAM ou fonction) d'une organisation ou d'un compte ont tenté pour la dernière fois d'accéder au service et à quel moment. Pour en savoir plus sur le rapport et sur la façon de consulter les dernières informations consultées pour AWS Organizations, voirAffichage des dernières informations consultées pour Organizations.

Pour obtenir des exemples de scénarios, afin d'utiliser les dernières informations consultées pour prendre des décisions concernant les autorisations que vous accordez à vos entités Organizations, consultez Exemples de scénarios d'utilisation des dernières informations consultées.

Choses à savoir sur les dernières informations consultées

Avant d'utiliser les dernières informations consultées d'un rapport dans le but de modifier les autorisations d'une identité IAM ou d'une entité Organizations, passez en revue les détails suivants concernant ces informations.

- Période de suivi : l'activité récente apparaît dans la console IAM dans un délai de quatre heures. La période de suivi pour les informations de service s’étend sur au moins 400 jours, en fonction de la date à laquelle le service a commencé à suivre les informations sur les actions. La période de suivi des information sur les actions Amazon S3 a commencé le 12 avril 2020. La période de suivi des actions Amazon EC2, IAM et Lambda a commencé le 7 avril 2021. La période de suivi pour

tous les autres services a débuté le 23 mai 2023. Pour obtenir la liste des services pour lesquels les dernières informations consultées relatives à une action sont disponibles, veuillez consulter Services et actions concernant les dernières informations consultées relatives à une action IAM. Pour plus d'informations concernant les régions dans lesquelles les dernières informations consultées relatives à une action sont disponibles, veuillez consulter Où AWS suit les dernières informations consultées.

- Tentatives signalées : les dernières données consultées par le service incluent toutes les tentatives d'accès à une AWS API, et pas seulement les tentatives réussies. Cela inclut toutes les tentatives effectuées à l' AWS Management Console aide de l' AWS API via l'un des SDK ou l'un des outils de ligne de commande. Une entrée inattendue dans les données relatives aux services consultés en dernier ne signifie pas que votre compte a été mis en danger, car la demande a pu être refusée. Référez-vous à vos CloudTrail journaux en tant que source officielle pour obtenir des informations sur tous les appels d'API et savoir s'ils ont été réussis ou s'ils ont été refusés.
- PassRole— L'`iam:PassRole`action n'est pas suivie et n'est pas incluse dans les dernières informations consultées sur l'action IAM.
- Dernières informations consultées relatives à une action : les dernières informations consultées relatives à une action sont disponibles pour les actions de gestion de service auxquelles des identités IAM ont accédé. Veuillez consulter la liste des services et de leurs actions pour lesquels les derniers accès relatifs à une action ont fourni des informations.

> **Note**
>
> Les dernières informations consultées relatives à une action ne sont pas disponibles pour les événements de données Amazon S3.

- Événements de gestion : IAM fournit des informations d'action pour les événements de gestion des services qui sont enregistrés par CloudTrail. Parfois, les événements CloudTrail de gestion sont également appelés opérations du plan de contrôle ou événements du plan de contrôle. Les événements de gestion fournissent une visibilité sur les opérations administratives effectuées sur les ressources de votre entreprise Compte AWS. Pour en savoir plus sur les événements de gestion dans CloudTrail, consultez la section Journalisation des événements de gestion dans le Guide de AWS CloudTrail l'utilisateur.
- Report owner (Propriétaire du rapport) : seul le principal qui génère un rapport peut afficher les détails de ce dernier. Cela signifie que lorsque vous consultez les informations contenues dans le AWS Management Console, vous devrez peut-être attendre qu'elles soient générées et chargées. Si vous utilisez l' AWS API AWS CLI or pour obtenir les détails du rapport, vos informations

d'identification doivent correspondre à celles du principal qui a généré le rapport. Si vous utilisez des informations d'identification temporaires pour un rôle ou un utilisateur fédéré, vous devez générer et récupérer le rapport au cours de la même session. Pour plus d'informations sur les principaux de session à rôle endossé, reportez-vous à la section AWS Éléments de politique JSON : Principal.

- Ressources IAM : les dernières informations consultées concernant IAM incluent les ressources IAM (rôles, utilisateurs, groupes d'utilisateurs et politiques) de votre compte. Les dernières informations consultées par les Organizations incluent les principaux (utilisateurs IAM, rôles IAM ou les Utilisateur racine d'un compte AWS) de l'entité Organizations spécifiée. Les tentatives non authentifiées sont exclues des dernières informations consultées.
- Types de politique IAM : les dernières informations consultées concernant IAM incluent les services qui sont autorisés par les politiques d'une identité IAM. Il s'agit de politiques attachées à un rôle ou attachées à un utilisateur directement ou via un groupe. L'accès autorisé par d'autres types de politique n'est pas inclus dans votre rapport. Les types de politique exclus comprennent les politiques basées sur les ressources, les listes de contrôle d'accès, les politiques de contrôle des services (SCP) AWS Organizations , les limites d'autorisations IAM et les politiques de sessions. Les autorisations fournies par les rôles liés au service sont définies par le service auquel ils sont liés et ne peuvent pas être modifiées dans IAM. Pour en savoir plus sur les rôles liés au service, consultez Utilisation des rôles liés à un service Pour en savoir plus sur les différents types de politique sont évalués pour autoriser ou refuser l'accès, consultez Logique d'évaluation de politiques.
- Organizations policy types (Types de politique Organizations) : les informations concernant AWS Organizations incluent uniquement les services qui sont autorisés par les politiques de contrôle de service (SCP) héritées d'une entité Organizations. Les SCP sont attachées à une racine, une unité organisationnelle ou un compte. L'accès autorisé par d'autres types de politique n'est pas inclus dans votre rapport. Les types de politique exclus comprennent les politiques basées sur une identité, les politiques basées sur les ressources, les listes de contrôle d'accès, les limites d'autorisations IAM et les politiques de sessions. Pour en savoir plus sur les différents types de politique qui sont évaluées pour autoriser ou refuser l'accès, veuillez consulter Logique d'évaluation de politiques.
- Spécification d'un ID de politique — Lorsque vous utilisez l' AWS API AWS CLI or pour générer un rapport sur les dernières informations consultées dans Organizations, vous pouvez éventuellement spécifier un ID de politique. Le rapport inclut des informations pour les services qui sont uniquement autorisées par cette politique. Les informations incluent la dernière activité de compte dans l'entité Organizations spécifiée ou ses enfants. Pour plus d'informations, consultez aws iam generate-organizations-access-report or GenerateOrganizationsAccessReport.

- Organizations management account (Compte de gestion de l'organisation) : vous devez vous connecter au compte de gestion de votre organisation pour afficher les dernières informations consultées sur le service. Vous pouvez choisir d'afficher les informations relatives au compte de gestion à l'aide de la console IAM, de l' AWS CLI API ou de l' AWS API. Le rapport qui en résulte répertorie tous les AWS services, car le compte de gestion n'est pas limité par les SCP. Si vous spécifiez un ID de politique dans l'interface de ligne de commande (CLI) ou l'API, la politique est ignorée. Pour chaque service, le rapport inclut uniquement les informations du compte de gestion. Toutefois, les rapports concernant les autres entités ne renvoient pas d'informations pour l'activité du compte principal.
- Organizations settings (Paramètres Organizations) : un administrateur doit activer les stratégies de contrôle de service (SCP) dans la racine de votre organisation pour que vous puissiez générer des données pour Organizations.

Autorisations nécessaires

Pour afficher les dernières informations consultées dans le AWS Management Console, vous devez disposer d'une politique qui accorde les autorisations nécessaires.

Autorisations pour des informations IAM

Pour utiliser la console IAM pour afficher les dernières informations consultées pour un utilisateur, un rôle ou une politique IAM, vous devez posséder une politique qui inclut les actions suivantes :

- `iam:GenerateServiceLastAccessedDetails`
- `iam:Get*`
- `iam:List*`

Ces autorisations permettent à l'utilisateur de voir les éléments suivants :

- Les utilisateurs, les groupes ou les rôles attachés à une politique gérée
- Les services auxquels un utilisateur ou un rôle peut accéder
- La dernière fois où ils se sont connectés au service
- La dernière fois qu'ils ont tenté d'utiliser une action Amazon EC2, IAM, Lambda ou Amazon S3 spécifique

Pour utiliser l' AWS API AWS CLI or afin de consulter les dernières informations consultées pour IAM, vous devez disposer d'autorisations correspondant à l'opération que vous souhaitez utiliser :

- `iam:GenerateServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetailsWithEntities`
- `iam:ListPoliciesGrantingServiceAccess`

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'affichage des dernières informations IAM consultées. En outre, il permet un accès en lecture seule à tous les éléments IAM. Cette politique définit des autorisations pour l'accès à la console et par programmation.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateServiceLastAccessedDetails",
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    }
```

Autorisations pour les informations AWS Organizations

Pour utiliser la console IAM pour afficher un rapport pour les entités racine, unité organisationnelle ou compte dans Organizations, vous devez avoir une politique qui inclut les actions suivantes :

- `iam:GenerateOrganizationsAccessReport`
- `iam:GetOrganizationsAccessReport`
- `organizations:DescribeAccount`
- `organizations:DescribeOrganization`
- `organizations:DescribeOrganizationalUnit`
- `organizations:DescribePolicy`
- `organizations:ListChildren`

- organizations:ListParents
- organizations:ListPoliciesForTarget
- organizations:ListRoots
- organizations:ListTargetsForPolicy

Pour utiliser l' AWS API AWS CLI or afin de consulter les dernières informations du service auxquelles les Organizations ont accédé, vous devez disposer d'une politique incluant les actions suivantes :

- iam:GenerateOrganizationsAccessReport
- iam:GetOrganizationsAccessReport
- organizations:DescribePolicy
- organizations:ListChildren
- organizations:ListParents
- organizations:ListPoliciesForTarget
- organizations:ListRoots
- organizations:ListTargetsForPolicy

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise l'affichage des dernières informations de service consultées pour Organizations. En outre, il permet un accès en lecture seule à tous les éléments Organizations. Cette politique définit des autorisations pour l'accès à la console et par programmation.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateOrganizationsAccessReport",
            "iam:GetOrganizationsAccessReport",
            "organizations:Describe*",
            "organizations:List*"
        ],
        "Resource": "*"
    }
}
```

Vous pouvez également utiliser la clé de OrganizationsPolicyId condition iam : pour autoriser la génération d'un rapport uniquement pour une politique d'Organizations spécifique. Pour un exemple de politique, consultez IAM : afficher les dernières informations consultées relatives au service pour une politique Organizations.

Activité de dépannage pour des entités IAM et Organizations

Dans certains cas, le AWS Management Console dernier tableau d'informations auquel vous avez accédé est peut-être vide. Ou peut-être que votre demande AWS CLI ou celle de AWS l'API renvoie un ensemble d'informations vide ou un champ nul. Dans ces cas, passez en revue les problèmes suivants :

- Pour les dernières informations consultées relatives à une action, cette dernière, que vous vous attendez à voir, peut ne pas être renvoyée dans la liste. Cela peut se produire soit parce que l'identité IAM n'est pas autorisée à effectuer l'action, soit parce AWS qu'elle n'assure pas encore le suivi de l'action pour les dernières informations consultées.
- Pour un utilisateur IAM, assurez-vous qu'il possède au moins une politique gérée ou une politique en ligne attachée, directement ou via l'appartenance à des groupes.
- Pour un groupe IAM, vérifiez qu'il possède au moins une politique gérée ou une politique en ligne attachée.
- Pour un groupe IAM, le rapport renvoie uniquement les dernières informations consultées relatives au service pour les membres ayant utilisé les politiques du groupe pour accéder à un service. Pour savoir si un membre a utilisé d'autres politiques, vérifiez les dernières informations consultées pour cet utilisateur.
- Pour un rôle IAM, vérifiez que le rôle possède au moins une politique gérée ou une politique en ligne attachée.
- Pour une entité IAM (utilisateur ou rôle), recherchez les autres types de politique qui peuvent affecter les autorisations de cette entité. Il s'agit notamment des politiques basées sur les ressources, des listes de contrôle d'accès, des AWS Organizations politiques, des limites d'autorisations IAM ou des politiques de session. Pour plus d’informations, consultez Types de politique ou Évaluation des politiques dans un compte unique.
- Dans le cas d'une politique IAM, assurez-vous que la politique gérée spécifiée est attachée à au moins un utilisateur, un groupe avec des membres ou un rôle.
- Pour une entité Organizations (racine, unité organisationnelle ou compte), assurez-vous que vous êtes connecté à l'aide des informations d'identification du compte de gestion Organizations.

- Vérifiez que les stratégies de contrôle de service (SCP) sont activées dans votre racine d'organisation.
- Les dernières informations consultées relatives à une action ne sont disponibles que pour les actions répertoriées dans Services et actions concernant les dernières informations consultées relatives à une action IAM.

Lorsque vous apportez des modifications, attendez au moins 4 heures avant que l'activité ne s'affiche dans votre console IAM. Si vous utilisez l' AWS API AWS CLI or, vous devez générer un nouveau rapport pour afficher les informations mises à jour.

Où AWS suit les dernières informations consultées

AWS collecte les dernières informations consultées pour les AWS régions standard. Lorsque AWS vous ajoutez des régions supplémentaires, celles-ci sont ajoutées au tableau suivant, y compris la date de AWS début du suivi des informations dans chaque région.

- Informations sur le service : la période de suivi des services correspond au moins à 400 jours, ou moins si votre région a commencé à suivre cette fonctionnalité au cours des 400 derniers jours.
- Information sur les actions : la période de suivi des actions de gestion Amazon S3 a commencé le 12 avril 2020. La période de suivi des actions de gestion Amazon EC2, IAM et Lambda a commencé le 7 avril 2021. La période de suivi des actions de gestion pour tous les autres services a débuté le 23 mai 2023. Si la date de suivi d'une région est postérieure au 23 mai 2023, les dernières informations consultées relatives à une action dans cette région débuteront à la date la plus tardive.

Nom de la région	Région	Date de début de suivi
USA Est (Ohio)	us-east-2	27 octobre 2017
US East (Virginie du Nord)	us-east-1	1er octobre 2015
USA Ouest (Californie du Nord)	us-west-1	1er octobre 2015
USA Ouest (Oregon)	us-west-2	1er octobre 2015
Afrique (Le Cap)	af-south-1	22 avril 2020

Nom de la région	Région	Date de début de suivi
Asie-Pacifique (Hong Kong)	ap-east-1	24 avril 2019
Asie-Pacifique (Hyderabad)	ap-south-2	22 novembre 2022
Asie-Pacifique (Jakarta)	ap-southeast-3	13 décembre 2021
Asie-Pacifique (Melbourne)	ap-southeast-4	23 janvier 2023
Asie-Pacifique (Mumbai)	ap-south-1	27 juin 2016
Asie-Pacifique (Osaka)	ap-northeast-3	11 février 2018
Asie-Pacifique (Séoul)	ap-northeast-2	6 janvier 2016
Asie-Pacifique (Singapour)	ap-southeast-1	1er octobre 2015
Asie-Pacifique (Sydney)	ap-southeast-2	1er octobre 2015
Asie-Pacifique (Tokyo)	ap-northeast-1	1er octobre 2015
Canada (Centre)	ca-central-1	28 octobre 2017
Europe (Francfort)	eu-central-1	1er octobre 2015
Europe (Irlande)	eu-west-1	1er octobre 2015
Europe (Londres)	eu-west-2	28 octobre 2017
Europe (Milan)	eu-south-1	28 avril 2020
Europe (Paris)	eu-west-3	18 décembre 2017
Europe (Espagne)	eu-south-2	15 novembre 2022
Europe (Stockholm)	eu-north-1	12 décembre 2018
Europe (Zurich)	eu-central-2	8 novembre 2022
Israël (Tel Aviv)	il-central-1	1er août 2023

Nom de la région	Région	Date de début de suivi
Moyen-Orient (Bahreïn)	me-south-1	29 juillet 2019
Moyen-Orient (EAU)	me-central-1	30 août 2022
Amérique du Sud (São Paulo)	sa-east-1	11 décembre 2015
AWS GovCloud (USA Est)	us-gov-east-1	1er juillet 2023
AWS GovCloud (US-Ouest)	us-gov-west-1	1er juillet 2023

Si une région n'est pas répertoriée dans le tableau précédent, cette région ne fournit pas encore les dernières informations consultées.

Une AWS région est un ensemble de AWS ressources dans une zone géographique. Les régions sont regroupées en partitions. Les régions standard sont les celles qui appartiennent à la partition `aws`. Pour de plus amples informations sur les différentes partitions, consultez Format des ARN (Amazon Resource Names) dans la Références générales AWS. Pour plus d'informations sur les régions, voir également À propos AWS des régions dans le Références générales AWS.

Affichage des dernières informations consultées pour IAM

Vous pouvez consulter les dernières informations consultées pour IAM à l'aide de l' AWS API AWS Management Console AWS CLI, ou. Veuillez consulter la liste des services et de leurs actions pour lesquels les dernières informations consultées sont affichées. Pour de plus amples informations sur les dernières informations consultées, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Vous pouvez consulter les informations relatives aux types de ressources suivants dans IAM. Dans chaque cas, les informations incluent les services autorisés pour la période donnée :

- Utilisateur : afficher la dernière fois que l'utilisateur a tenté d'accéder à chaque service autorisé.
- User group (Groupe) : consultez les informations sur la dernière fois où un membre du groupe a tenté d'accéder à chaque service autorisé. Ce rapport inclut également le nombre total de membres qui ont tenté un accès.
- Role (Rôle) : affichez la dernière fois où quelqu'un a utilisé le rôle pour tenter d'accéder à chaque service autorisé.

- Policy (Politique) : consultez les informations sur la dernière fois où un utilisateur ou un rôle a tenté d'accéder à chaque service autorisé. Ce rapport inclut également le nombre total d'entités qui ont tenté un accès.

> **Note**
>
> Avant de consulter les informations d'accès d'une ressource IAM, vous devez bien comprendre la période couverte par le rapport, les entités rapportées et les types de politique évalués pour vos informations. Pour en savoir plus, consultez the section called “Choses à savoir sur les dernières informations consultées”.

Affichage des informations pour IAM (console)

Vous pouvez afficher les dernières informations consultées pour dans l'onglet Access Advisor de la console IAM.

Pour afficher les informations pour IAM (console)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez User groups (Groupes d'utilisateurs), Users (Utilisateurs), Roles (Rôles) ou Policies (Politiques).
3. Choisissez un nom d'utilisateur, de groupe d'utilisateurs, de rôle ou de politique pour ouvrir la page Summary (Résumé) et sélectionnez l'onglet Access Advisor. Affichez les informations suivantes, en fonction de la ressource que vous avez choisie :

   - User group (Groupe d'utilisateurs) : affiche la liste des services auxquels les membres du groupe d'utilisateurs peuvent accéder. Vous pouvez également afficher la dernière fois qu'un membre a accédé au service, quelles politiques de groupe d'utilisateurs il a utilisées et quel membre du groupe d'utilisateurs a effectué la demande. Choisissez le nom de la politique pour savoir s'il s'agit d'une politique gérée ou d'une politique de groupe d'utilisateurs en ligne. Choisissez le nom du membre du groupe pour afficher tous les membres du groupe d'utilisateurs et quand ils ont consulté pour la dernière fois le service.
   - User (Utilisateur) : affiche la liste des services auxquels l'utilisateur peut accéder. Vous pouvez également afficher la date à laquelle ils ont accédé au service pour la dernière fois et les politiques qui sont actuellement associées à l'utilisateur. Choisissez le nom de la politique pour

savoir s'il s'agit d'une politique gérée, d'une politique d'utilisateur en ligne ou d'une politique en ligne pour le groupe.

- Role (Rôle) : affichez la liste des services auxquels le rôle peut accéder, à quel moment le rôle a accédé au service pour la dernière fois et quelles politiques ont été utilisées. Choisissez le nom de la politique pour savoir s'il s'agit d'une politique gérée ou d'une politique de rôle en ligne.
- Policy (Politique) : affiche la liste des services avec des actions autorisées dans la politique. Vous pouvez également afficher la dernière fois où la politique a été utilisée pour accéder au service et quelle entité (utilisateur ou rôle) l'a utilisé. La date Dernier accès inclut également le moment où l'accès est accordé à cette politique par l'intermédiaire d'une autre politique. Choisissez le nom de l'entité pour connaître les entités ayant cette politique attachée et quand elles ont consulté pour la dernière fois le service.

4. Dans la colonne Service du tableau, choisissez le nom de l'un des services qui inclut les dernières informations consultées relatives à une action pour afficher une liste des actions de gestion auxquelles les entités IAM ont tenté d'accéder. Vous pouvez afficher l' Région AWS et un horodatage indiquant la dernière fois que quelqu'un a tenté d'exécuter l'action.
5. La colonne Dernier accès s'affiche pour les services et les actions de gestion des services qui incluent les dernières informations consultées relatives à une action. Examinez les résultats possibles suivants qui sont renvoyés dans cette colonne. Ces résultats varient selon qu'un service ou une action est autorisé, qu'il a été consulté et qu'il est suivi ou non par AWS les dernières informations consultées.

   il y a <nombre de> jours

   Nombre de jours écoulés depuis que le service ou l'action a été utilisé pendant la période de suivi. La période de suivi des services s'étend sur les 400 derniers jours. La période de suivi des actions Amazon S3 a commencé le 12 avril 2020. La période de suivi des actions Amazon EC2, IAM et Lambda a commencé le 7 avril 2021. La période de suivi pour tous les autres services a débuté le 23 mai 2023. Pour en savoir plus sur les dates de début du suivi pour chacune d'entre elles Région AWS, consultezOù AWS suit les dernières informations consultées.

   Aucun accès pendant la période de suivi

   Le service ou l'action suivi n'a pas été utilisé par une entité au cours de la période de suivi.

Vous pouvez disposez d'autorisations pour une action qui n'apparaît pas dans la liste. C'est le cas lorsque les informations de suivi de l'action ne sont pas actuellement prises en compte par AWS. Vous ne devez pas prendre de décisions sur les autorisations uniquement en raison de l'absence d'informations de suivi. Nous vous recommandons plutôt d'utiliser ces informations pour éclairer et fonder votre stratégie globale d'octroi du moindre privilège. Vérifiez vos politiques pour confirmer que le niveau d'accès est approprié.

Affichage des informations pour IAM (AWS CLI)

Vous pouvez utiliser le AWS CLI pour récupérer des informations sur la dernière fois qu'une ressource IAM a été utilisée pour tenter d'accéder à des AWS services et à des actions Amazon S3, Amazon EC2, IAM et Lambda. Une ressource IAM peut être un utilisateur, un groupe d'utilisateurs, un rôle ou une politique.

Pour afficher les informations pour IAM (AWS CLI)

1. Générez un rapport. La demande doit inclure l'ARN de la ressource IAM (utilisateur, groupe d'utilisateurs, rôle ou politique) pour lequel vous voulez un rapport. Vous pouvez spécifier le niveau de granularité que vous souhaitez générer dans le rapport pour afficher les détails sur l'accès pour les services ou pour les services et les actions. La demande renvoie un `job-id` que vous pouvez ensuite utiliser dans les opérations `get-service-last-accessed-details-with-entities` et `get-service-last-accessed-details` pour surveiller l'état `job-status` jusqu'à ce que la tâche soit terminée.

   - aws iam -détails generate-service-last-accessed
2. Récupérez les informations sur le rapport à l'aide du paramètre `job-id` de l'étape précédente.

   - aws iam -détails get-service-last-accessed

   Cette opération renvoie les informations suivantes, en fonction du type de ressource et du niveau de granularité que vous avez demandé dans l'opération `generate-service-last-accessed-details` :

   - User (Utilisateur) : renvoie une liste de services auxquels l'utilisateur spécifié peut accéder. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative de l'utilisateur, ainsi que l'ARN de l'utilisateur.

- Groupe d'utilisateurs : renvoie la liste des services auxquels les membres du groupe d'utilisateurs spécifié peuvent accéder à l'aide des politiques attachées au groupe d'utilisateurs. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative effectuée par un membre du groupe d'utilisateurs. Il renvoie également l'ARN de cet utilisateur et le nombre total de membres du groupe d'utilisateurs qui ont tenté d'accéder au service. Utilisez cette GetServiceLastAccessedDetailsWithEntitiesopération pour récupérer la liste de tous les membres.
- Role (Rôle) : renvoie la liste des services auxquels le rôle spécifié peut accéder. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative du rôle, ainsi que l'ARN du rôle.
- Policy (Politique) : renvoie la liste des services pour lesquels la politique spécifiée autorise l'accès. Pour chaque service, l'opération renvoie la date et l'heure auxquelles une entité (utilisateur ou rôle) a tenté pour la dernière fois d'accéder au service à l'aide de la politique. Elle renvoie également l'ARN de cette entité et le nombre total d'entités ayant tenté l'accès.

3. En savoir plus sur les entités qui ont utilisé les autorisations de groupe d'utilisateurs ou de politique lors de la tentative d'accès à un service spécifique. Cette opération renvoie la liste des entités avec l'ARN, l'ID, le nom, le chemin, le type (utilisateur ou rôle) de chaque entité, et la date à laquelle elles ont tenté pour la dernière fois d'accéder au service. Vous pouvez également utiliser cette opération pour les utilisateurs et les rôles, mais elle ne renvoie que les informations relatives à cette entité.

   - c'est moi - get-service-last-accessed details-with-entities

4. En savoir plus sur les politiques basées sur une identité qu'une identité (utilisateur, groupe d'utilisateurs ou rôle) a utilisées lors de la tentative d'accès à un service spécifique. Lorsque vous spécifiez une identité et un service, cette opération renvoie une liste de stratégies d'autorisation que l'identité peut utiliser pour accéder au service spécifié. Cette opération donne l'état actuel des stratégies et ne dépend pas du rapport généré. Elle ne retourne pas non plus d'autres types de politiques, tels que les politiques basées sur les ressources, les listes de contrôle d'accès, les politiques AWS Organizations , les limites d'autorisations IAM ou les politiques de sessions. Pour plus d’informations, consultez Types de politique ou Évaluation des politiques dans un compte unique.

   - aws iam -access list-policies-granting-service

Affichage des informations pour IAM (AWS API)

Vous pouvez utiliser l' AWS API pour récupérer des informations sur la dernière fois qu'une ressource IAM a été utilisée pour tenter d'accéder à des AWS services et à des actions Amazon S3, Amazon EC2, IAM et Lambda. Une ressource IAM peut être un utilisateur, un groupe d'utilisateurs, un rôle ou une politique. Vous pouvez spécifier le niveau de granularité à générer dans le rapport pour afficher les détails des services ou des services et des actions.

Pour consulter les informations relatives à l'IAM (AWS API)

1. Générez un rapport. La demande doit inclure l'ARN de la ressource IAM (utilisateur, groupe d'utilisateurs, rôle ou politique) pour lequel vous voulez un rapport. Elle renvoie un `JobId` que vous pouvez ensuite utiliser dans les opérations `GetServiceLastAccessedDetailsWithEntities` et `GetServiceLastAccessedDetails` pour surveiller l'état `JobStatus` jusqu'à ce que la tâche soit terminée.

   - GenerateServiceLastAccessedDetails
2. Récupérez les informations sur le rapport à l'aide du paramètre `JobId` de l'étape précédente.

   - GetServiceLastAccessedDetails

   Cette opération renvoie les informations suivantes, en fonction du type de ressource et du niveau de granularité que vous avez demandé dans l'opération `GenerateServiceLastAccessedDetails` :

   - User (Utilisateur) : renvoie une liste de services auxquels l'utilisateur spécifié peut accéder. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative de l'utilisateur, ainsi que l'ARN de l'utilisateur.
   - Groupe d'utilisateurs : renvoie la liste des services auxquels les membres du groupe d'utilisateurs spécifié peuvent accéder à l'aide des politiques attachées au groupe d'utilisateurs. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative effectuée par un membre du groupe d'utilisateurs. Il renvoie également l'ARN de cet utilisateur et le nombre total de membres du groupe d'utilisateurs qui ont tenté d'accéder au service. Utilisez cette GetServiceLastAccessedDetailsWithEntitiesopération pour récupérer la liste de tous les membres.

- Role (Rôle) : renvoie la liste des services auxquels le rôle spécifié peut accéder. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative du rôle, ainsi que l'ARN du rôle.
- Policy (Politique) : renvoie la liste des services pour lesquels la politique spécifiée autorise l'accès. Pour chaque service, l'opération renvoie la date et l'heure auxquelles une entité (utilisateur ou rôle) a tenté pour la dernière fois d'accéder au service à l'aide de la politique. Elle renvoie également l'ARN de cette entité et le nombre total d'entités ayant tenté l'accès.

3. En savoir plus sur les entités qui ont utilisé les autorisations de groupe d'utilisateurs ou de politique lors de la tentative d'accès à un service spécifique. Cette opération renvoie la liste des entités avec l'ARN, l'ID, le nom, le chemin, le type (utilisateur ou rôle) de chaque entité, et la date à laquelle elles ont tenté pour la dernière fois d'accéder au service. Vous pouvez également utiliser cette opération pour les utilisateurs et les rôles, mais elle ne renvoie que les informations relatives à cette entité.

   - GetServiceLastAccessedDetailsWithEntities

4. En savoir plus sur les politiques basées sur une identité qu'une identité (utilisateur, groupe d'utilisateurs ou rôle) a utilisées lors de la tentative d'accès à un service spécifique. Lorsque vous spécifiez une identité et un service, cette opération renvoie une liste de stratégies d'autorisation que l'identité peut utiliser pour accéder au service spécifié. Cette opération donne l'état actuel des stratégies et ne dépend pas du rapport généré. Elle ne retourne pas non plus d'autres types de politiques, tels que les politiques basées sur les ressources, les listes de contrôle d'accès, les politiques AWS Organizations , les limites d'autorisations IAM ou les politiques de sessions. Pour plus d'informations, consultez Types de politique ou Évaluation des politiques dans un compte unique.

   - ListPoliciesGrantingServiceAccess

Affichage des dernières informations consultées pour Organizations

Vous pouvez consulter les informations du dernier accès au service à AWS Organizations l'aide de la console IAM ou de AWS l'API. AWS CLI Pour connaître les informations importantes sur les données, les autorisations requises, la résolution de problèmes et les régions prises en charge, veuillez consulter Affiner les autorisations en AWS utilisant les dernières informations consultées.

Lorsque vous vous connectez à la console IAM à l'aide AWS Organizations des informations d'identification du compte de gestion, vous pouvez consulter les informations relatives à n'importe quelle entité de votre organisation. Les entités Organizations comprennent la racine de l'organisation,

les unités d'organisation (OU) et les comptes. Vous pouvez également utiliser la console IAM pour afficher des informations relatives aux politiques de contrôle des services (SCP) de votre organisation. IAM affiche une liste des services qui sont autorisés par chacune des SCP qui s'appliquent à l'entité. Pour chaque service, vous pouvez afficher les informations d'activité du compte les plus récentes pour l'entité Organizations choisie ou les enfants de l'entité.

Lorsque vous utilisez l' AWS API AWS CLI or avec les informations d'identification du compte de gestion, vous pouvez générer un rapport pour toutes les entités ou politiques de votre organisation. Un rapport de données par programmation pour une entité comprend une liste des services qui sont autorisés par les stratégies de contrôle de service qui s'appliquent à l'entité. Pour chaque service, le rapport inclut l'activité la plus récente pour les comptes de l'entité Organizations spécifiée ou de la sous-arborescence de l'entité.

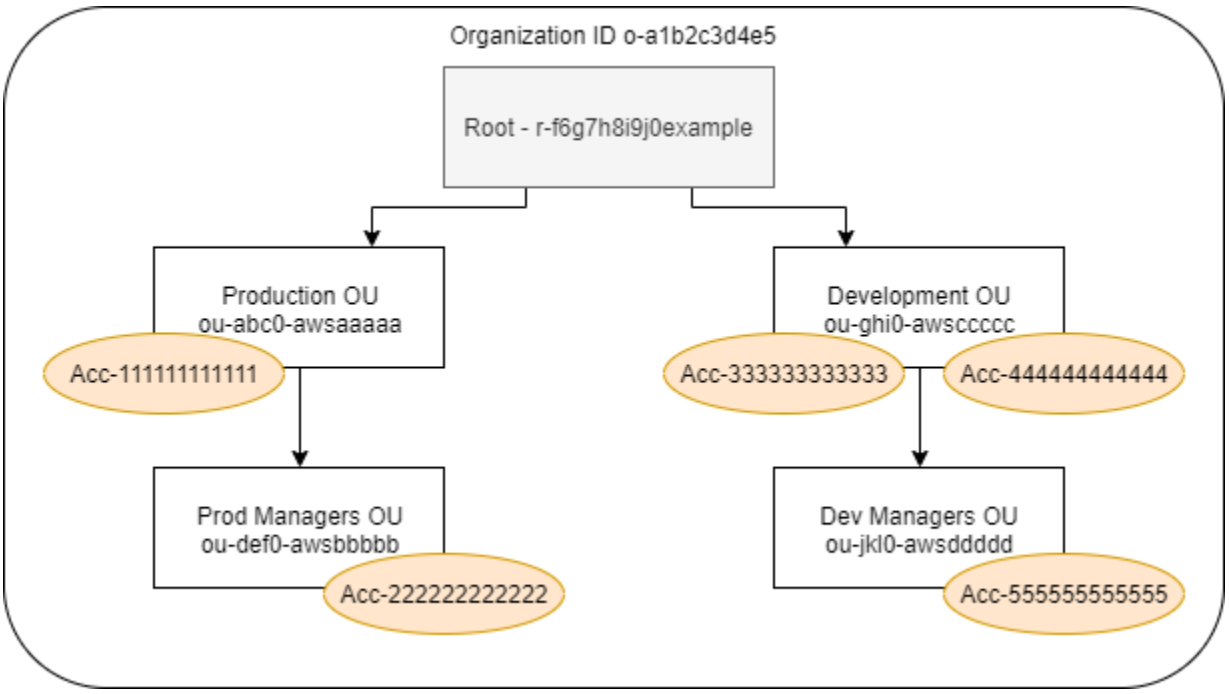
Lorsque vous générez un rapport d'informations par programmation pour une politique, vous devez spécifier une entité . Ce rapport inclut une liste des services qui sont autorisés par la stratégie de contrôle de service spécifiée. Pour chaque service, il inclut la dernière activité du compte dans l'entité ou dans les enfants de l'entité qui sont autorisés par cette politique. Pour plus d'informations, consultez aws iam generate-organizations-access-report or GenerateOrganizationsAccessReport.

Avant d'afficher le rapport, assurez-vous de comprendre les exigences et les informations du compte de gestion, la période couverte par le rapport, les entités concernées par le rapport, ainsi que les types de politiques évaluées. Pour en savoir plus, consultez the section called “Choses à savoir sur les dernières informations consultées”.

Comprendre le chemin d'entité AWS Organizations

Lorsque vous utilisez l' AWS API AWS CLI or pour générer un rapport d' AWS Organizations accès, vous devez spécifier un chemin d'entité. Un chemin est une représentation textuelle de la structure d'une entité Organizations.

Vous pouvez créer un chemin d'entité à l'aide de la structure connue de votre organisation. Supposons, par exemple, que vous disposiez de la structure organisationnelle suivante dans AWS Organizations.



Le chemin de l'unité d'organisation Dev Managers est créé à l'aide des ID de l'organisation, de la racine et de toutes les unités d'organisation du chemin jusqu'à l'unité d'organisation incluse.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-ghi0-awsccccc/ou-jkl0-awsdddddd/
```

Le chemin du compte dans l'unité d'organisation Production est généré à l'aide des ID de l'organisation, de la racine, de l'unité d'organisation et du numéro de compte.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-abc0-awsaaaaa/111111111111/
```

> **Note**
>
> Les ID d'organisation sont globalement uniques, mais les ID d'unité d'organisation et les ID racine ne sont uniques qu'au sein d'une organisation. Cela signifie qu'aucune organisation ne partage le même ID d'organisation. Toutefois, une autre organisation peut avoir le même ID d'unité d'organisation ou ID racine que vous. Nous vous recommandons de toujours inclure l'ID d'organisation lorsque vous spécifiez une unité d'organisation ou une racine.

Affichage des informations pour Organizations (console)

Vous pouvez utiliser la console IAM pour afficher les informations relatives aux derniers services consultés pour votre compte racine, une unité organisationnelle ou une politique.

Pour afficher les informations relatives à la racine (console)

1. Connectez-vous à l' AWS Management Console aide des informations d'identification du compte de gestion Organizations et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation situé sous la section Access reports (Rapports d'accès), choisissez Organization activity (Activité de l'organisation).
3. Dans la page Organization activity (Activité de l'organisation), choisissez Root (Racine).
4. Sur l'onglet Details and activity (Détails et activité), affichez la section Service access report (Rapport d'accès aux services). Les informations comprennent une liste des services autorisés par les politiques qui sont attachées directement à la racine. Les informations vous indiquent également le compte à partir duquel le service a été consulté pour la dernière fois et le moment auquel cette consultation a eu lieu. Pour de plus amples informations sur le principal ayant consulté le service, connectez-vous en tant qu'administrateur à ce compte et affichez les dernières informations consultées relatives au service IAM.
5. Sélectionnez l'onglet Attached SCPs (SCP attachés) pour afficher la liste des politiques de contrôle des services (SCP) qui sont attachées à la racine. IAM vous indique le nombre d'entités cibles auxquelles chaque politique est attachée. Vous pouvez utiliser ces informations pour déterminer la stratégie de contrôle de service à passer en revue.
6. Choisissez le nom d'une politique de contrôle de service pour afficher tous les services qu'elle autorise. Pour chaque service, affichez le compte à partir duquel le service a été consulté pour la dernière fois et le moment auquel cette consultation a eu lieu.
7. Sélectionnez Edit in AWS Organizations (Modifier dans AO) pour afficher des détails supplémentaires et modifier la stratégie de contrôle de service dans la console Organizations. Pour plus d'information, consultez la section relative aux mises à jour Over-the-Air dans le AWS Organizations User Guide.

Pour afficher les informations d'une unité d'organisation ou d'un compte (console)

1. Connectez-vous à l' AWS Management Console aide des informations d'identification du compte de gestion Organizations et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation situé sous la section Access reports (Rapports d'accès), choisissez Organization activity (Activité de l'organisation).
3. Sur la page Organization activity (Activité de l'organisation), développez la structure de votre organisation. Ensuite, choisissez le nom de l'unité organisationnelle ou du compte que vous souhaitez afficher (à l'exception du compte de gestion).
4. Sur l'onglet Details and activity (Détails et activité), affichez la section Service access report (Rapport d'accès aux services). Les informations comprennent une liste des services qui sont autorisés par les stratégies de contrôle de service attachées à l'unité organisationnelle ou au compte et à tous ses parents. Les informations vous indiquent également le compte à partir duquel le service a été consulté pour la dernière fois et le moment auquel cette consultation a eu lieu. Pour de plus amples informations sur le principal ayant consulté le service, connectez-vous en tant qu'administrateur à ce compte et affichez les dernières informations consultées relatives au service IAM.
5. Sélectionnez l'onglet Attached SCPs (SCP attachés) pour afficher la liste des politiques de contrôle des services (SCP) qui sont attachées directement à l'UO ou au compte. IAM vous indique le nombre d'entités cibles auxquelles chaque politique est attachée. Vous pouvez utiliser ces informations pour déterminer la stratégie de contrôle de service à passer en revue.
6. Choisissez le nom d'une politique de contrôle de service pour afficher tous les services qu'elle autorise. Pour chaque service, affichez le compte à partir duquel le service a été consulté pour la dernière fois et le moment auquel cette consultation a eu lieu.
7. Sélectionnez Edit in AWS Organizations (Modifier dans AO) pour afficher des détails supplémentaires et modifier la stratégie de contrôle de service dans la console Organizations. Pour plus d'information, consultez la section relative aux mises à jour Over-the-Air dans le AWS Organizations User Guide.

Pour afficher les informations du compte de gestion (console)

1. Connectez-vous à l' AWS Management Console aide des informations d'identification du compte de gestion Organizations et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.

2. Dans le panneau de navigation situé sous la section Access reports (Rapports d'accès), choisissez Organization activity (Activité de l'organisation).
3. Sur la page Organization activity (Activité de l'organisation), développez la structure de votre organisation et choisissez le nom de votre compte de gestion.
4. Sur l'onglet Details and activity (Détails et activité), affichez la section Service access report (Rapport d'accès aux services). Les informations incluent une liste de tous les services AWS . Le compte de gestion n'est pas limité par les SCP. Les informations vous indiquent si le compte a consulté le service dernièrement et le moment auquel cette consultation s'est produite. Pour de plus amples informations sur le principal ayant consulté le service, connectez-vous en tant qu'administrateur à ce compte et affichez les dernières informations consultées relatives au service IAM.
5. Sélectionnez l'onglet Attached SCPs (Stratégies de contrôle des services attachées) pour vérifier qu'il n'y a pas de stratégies de contrôle de service attachées, car le compte est le compte de gestion.

Pour afficher les informations d'une politique (console)

1. Connectez-vous à l' AWS Management Console aide des informations d'identification du compte de gestion Organizations et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation situé sous la section Access reports (Rapports d'accès), choisissez Service control policies (SCPs) (Politiques de contrôle des services (SCP)).
3. Dans la page Service control policies (SCPs) (Politiques de contrôle de service (SCP)), affichez la liste des politiques de votre organisation. Vous pouvez consulter le nombre d'entités cibles auxquelles chaque politique est attachée.
4. Choisissez le nom d'une politique de contrôle de service pour afficher tous les services qu'elle autorise. Pour chaque service, affichez le compte à partir duquel le service a été consulté pour la dernière fois et le moment auquel cette consultation a eu lieu.
5. Sélectionnez Edit in AWS Organizations (Modifier dans AO) pour afficher des détails supplémentaires et modifier la stratégie de contrôle de service dans la console Organizations. Pour plus d'information, consultez la section relative aux mises à jour Over-the-Air dans le AWS Organizations User Guide.

Affichage des informations pour Organizations (AWS CLI)

Vous pouvez utiliser le AWS CLI pour récupérer les dernières informations auxquelles vous avez accédé au service pour la racine, l'unité d'organisation, le compte ou la politique de votre organisation.

Pour afficher les dernières informations consultées relatives au service Organizations (AWS CLI)

1. Utilisez vos informations d'identification de compte de gestion Organizations avec les autorisations IAM et Organizations requises, puis vérifiez que les stratégies de contrôle de service sont activées pour votre racine. Pour plus d'informations, veuillez consulter Choses à savoir sur les dernières informations consultées.
2. Générez un rapport. La demande doit inclure le chemin d'accès de l'entité Organizations (racine, unité organisationnelle ou compte) pour laquelle vous voulez un rapport. Si vous le souhaitez, vous pouvez inclure un paramètre `organization-policy-id` pour afficher un rapport pour une politique spécifique. Cette commande renvoie un élément `job-id` que vous pouvez ensuite utiliser dans la commande `get-organizations-access-report` pour surveiller l'élément `job-status` jusqu'à ce que la tâche soit terminée.

   - était un objectif generate-organizations-access-report
3. Récupérez les informations sur le rapport à l'aide du paramètre `job-id` de l'étape précédente.

   - était un objectif get-organizations-access-report

   Cette commande renvoie une liste des services auxquels les membres de l'entité peuvent accéder. Pour chaque service, la commande renvoie la date et l'heure de la dernière tentative d'un membre du compte et le chemin d'entité du compte. Elle renvoie également le nombre total de services qui sont disponibles en accès et le nombre de services qui n'ont pas été consultés. Si vous avez spécifié le paramètre facultatif `organizations-policy-id`, les services qui sont disponibles en accès sont ceux qui sont autorisés par la politique spécifiée.

Affichage des informations pour les Organizations (AWS API)

Vous pouvez utiliser l' AWS API pour récupérer les dernières informations de service auxquelles vous avez accédé pour la racine, l'unité d'organisation, le compte ou la politique de votre organisation.

Pour consulter les dernières informations consultées par le service Organizations (AWS API)

1. Utilisez vos informations d'identification de compte de gestion Organizations avec les autorisations IAM et Organizations requises, puis vérifiez que les stratégies de contrôle de service sont activées pour votre racine. Pour plus d'informations, veuillez consulter Choses à savoir sur les dernières informations consultées.
2. Générez un rapport. La demande doit inclure le chemin d'accès de l'entité Organizations (racine, unité organisationnelle ou compte) pour laquelle vous voulez un rapport. Si vous le souhaitez, vous pouvez inclure un paramètre `OrganizationsPolicyId` pour afficher un rapport pour une politique spécifique. Cette opération renvoie un élément `JobId` que vous pouvez ensuite utiliser dans l'opération `GetOrganizationsAccessReport` pour surveiller l'état `JobStatus` jusqu'à ce que la tâche soit terminée.

   - GenerateOrganizationsAccessReport
3. Récupérez les informations sur le rapport à l'aide du paramètre `JobId` de l'étape précédente.

   - GetOrganizationsAccessReport

   Cette opération renvoie une liste des services auxquels les membres de l'entité peuvent accéder. Pour chaque service, l'opération renvoie la date et l'heure de la dernière tentative d'un membre du compte et le chemin d'entité du compte. Elle renvoie également le nombre total de services qui sont disponibles en accès et le nombre de services qui n'ont pas été consultés. Si vous avez spécifié le paramètre facultatif `OrganizationsPolicyId`, les services qui sont disponibles en accès sont ceux qui sont autorisés par la politique spécifiée.

Exemples de scénarios d'utilisation des dernières informations consultées

Vous pouvez utiliser les dernières informations consultées pour prendre des décisions concernant les autorisations que vous accordez à vos entités ou AWS Organizations entités IAM. Pour plus d'informations, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

> **Note**
>
> Avant de consulter les informations d'accès relatives à une entité ou à une politique dans IAM AWS Organizations, assurez-vous de bien comprendre la période de reporting, les entités

signalées et les types de politiques évalués pour vos données. Pour en savoir plus, consultez the section called “Choses à savoir sur les dernières informations consultées”.

En tant qu'administrateur, il vous appartient d'équilibrer l'accessibilité et le principe du moindre privilège requis pour votre entreprise.

Utilisation des informations pour réduire les autorisations d'un groupe IAM

Vous pouvez utiliser les dernières informations consultés pour réduire les autorisations de groupe IAM et inclure uniquement les services dont vos utilisateurs ont besoin. Cette méthode est une étape importante dans l'attribution du moindre privilège au niveau service.

Par exemple, Paulo Santos est l'administrateur chargé de définir les autorisations des AWS utilisateurs pour Example Corp. Cette société vient de commencer à utiliser AWS, et l'équipe de développement logiciel n'a pas encore défini les AWS services qu'elle utilisera. Paulo souhaite accorder à l'équipe l'autorisation d'accéder uniquement aux services dont elle a besoin, mais comme cela n'est pas encore défini, il leur attribue temporairement les autorisations utilisateur. Ensuite, il utilise les dernières informations consultées pour réduire les autorisations du groupe.

Paulo crée une politique gérée nommée `ExampleDevelopment` à l'aide du texte JSON suivant. Puis, il l'attache à un groupe appelé `Development` et ajoute tous les développeurs au groupe.

Note

Les utilisateurs avancés de Paulo peuvent avoir besoin d'autorisations `iam:CreateServiceLinkedRole` pour utiliser certains services et fonctionnalités. Il comprend que l'ajout de cette autorisation permet aux utilisateurs de créer n'importe quel rôle lié au service. Il accepte ce risque pour ses utilisateurs avancés.

```
{

    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToAllServicesExceptPeopleManagement",
            "Effect": "Allow",
            "NotAction": [
```

```
                "iam:*",
                "organizations:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

Paulo décide d'attendre 90 jours avant d'afficher les dernières informations consultées pour le groupe `Development` à l'aide de la AWS Management Console. Il affiche la liste des services que les membres du groupe ont consultés. Il apprend que les utilisateurs ont accédé à cinq services la semaine dernière : AWS CloudTrail Amazon CloudWatch Logs, Amazon EC2 et Amazon S3. AWS KMS Ils ont eu accès à quelques autres services lors de leur première évaluation AWS, mais pas depuis.

Paulo décide pour réduire les autorisations de la politique de façon à inclure uniquement ces cinq services et les actions IAM et Organizations requises. Il modifie la politique `ExampleDevelopment` à l'aide du texte JSON suivant.

> **Note**
>
> Les utilisateurs avancés de Paulo peuvent avoir besoin d'autorisations `iam:CreateServiceLinkedRole` pour utiliser certains services et fonctionnalités. Il comprend que l'ajout de cette autorisation permet aux utilisateurs de créer n'importe quel rôle lié au service. Il accepte ce risque pour ses utilisateurs avancés.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "FullAccessToListedServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "kms:*",
                "cloudtrail:*",
                "logs:*",
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour réduire davantage les autorisations, Paulo peut consulter les événements du compte dans l'historique des AWS CloudTrail événements. Là, il peut afficher les informations détaillées des événements qu'il peut utiliser pour réduire les autorisations de la politique et inclure uniquement les actions et les ressources dont les développeurs ont besoin. Pour plus d'informations, consultez la section Affichage CloudTrail des événements dans la CloudTrail console dans le guide de AWS CloudTrail l'utilisateur.

Utilisation des informations pour réduire les autorisations d'un utilisateur IAM

Vous pouvez utiliser les dernières informations consultées pour réduire les autorisations d'un utilisateur IAM individuel.

Par exemple, Martha Rivera est une administratrice informatique chargée de veiller à ce que les membres de son entreprise ne disposent pas d' AWS autorisations excessives. Dans le cadre d'un contrôle de sécurité régulier, elle passe en revue les autorisations de tous les utilisateurs IAM. L'un des ces utilisateurs est un développeur d'applications nommé Nikhil Jayashankar, qui occupait précédemment le rôle d'ingénieur sécurité. Dans la mesure où ses tâches ont évolué, Nikhil est à

la fois membre du groupe app-dev et du groupe security-team. Le groupe app-dev pour son nouveau poste octroie des autorisations à plusieurs services, notamment Amazon EC2, Amazon EBS, Auto Scaling, Amazon S3, Route 53 et Elastic Transcoder. Le security-team groupe chargé de son ancien travail accorde des autorisations à IAM et CloudTrail.

En tant qu'administrateur, Martha se connecte à la console IAM, sélectionnez Users (Utilisateurs), le nom nikhilj, puis l'onglet Access Advisor.

Martha passe en revue la colonne Last Acceded et remarque que Nikhil n'a pas récemment accédé à IAM CloudTrail, Route 53, Amazon Elastic Transcoder et à un certain nombre d'autres services. AWS Nikhil a accédé à Amazon S3. Martha choisit S3 dans la liste des services et apprend que Nikhil a effectué quelques actions List S3 au cours des deux dernières semaines. Au sein de son entreprise, Martha confirme que Nikhil n'a CloudTrail plus besoin d'accéder à IAM pour des raisons professionnelles, car il n'est plus membre de l'équipe de sécurité interne.

Martha est maintenant prête à agir sur le service et des dernières information consultées relatives à l'action. Toutefois, à la différence du groupe de l'exemple précédent, un utilisateur IAM comme nikhilj peut être soumis à plusieurs politiques et appartenir à plusieurs groupes. Martha doit procéder avec précaution pour éviter de modifier par inadvertance l'accès de nikhilj ou d'autres membres du groupe. En plus d'apprendre quel accès Nikhil doit avoir, elle doit déterminer comment ces autorisations lui sont accordées.

Martha choisit l'onglet Autorisations, où elle consulte les politiques attachées directement à nikhilj et celles attachées à partir d'un groupe. Elle développe chaque politique et affiche le récapitulatif de la politique pour savoir quelle politique autorise l'accès aux services que Nikhil n'utilise pas :

- IAM — La politique IAMFullAccess AWS gérée est attachée nikhilj et attachée directement au security-team groupe.
- CloudTrail — La politique AWS CloudTrailReadOnlyAccess AWS gérée est attachée au security-team groupe.
- Route 53 : la politique App-Dev-Route53 gérée par le client est attachée au groupe app-dev.
- Elastic Transcoder : la politique App-Dev-ElasticTranscoder gérée par le client est attachée au groupe app-dev.

Martha décide de supprimer la politique IAMFullAccess AWS gérée qui est directement attachée ànikhilj. Elle supprime également l'appartenance de Nikhil au groupe security-team. Ces deux actions suppriment l'accès inutile à IAM et CloudTrail.

Les autorisations de Nikhil d'accéder à Route 53 et Elastic Transcoder sont octroyées par le groupe `app-dev`. Bien que Nikhil n'utilise pas ces services, il se peut que d'autres membres le fassent. Martha examine les dernières informations consultées pour le groupe `app-dev` et apprend que plusieurs membres ont récemment accédé à Route 53 et Amazon S3. Mais aucun membre du groupe n'a accédé à Elastic Transcoder au cours de la dernière année. Supprime du groupe la politique `App-Dev-ElasticTranscoder` gérée par le client.

Martha passe ensuite en revue les dernières informations consultées pour la politique `App-Dev-ElasticTranscoder` gérée par le client. Elle apprend que la politique n'est pas attachée à d'autres identités IAM. Elle vérifie au sein de son entreprise que la politique ne sera pas nécessaire à l'avenir, puis elle la supprime.

Utilisation des informations avant de supprimer des ressources IAM

Vous pouvez utiliser les dernières informations consultées avant de supprimer une ressource IAM, afin de vous assurer qu'un certain délai s'est écoulé depuis qu'une personne a utilisé la ressource pour la dernière fois. Cela s'applique aux utilisateurs, groupes, rôles et politiques. Pour de plus amples informations sur ces actions, veuillez consulter les rubriques suivantes :

- Users (Utilisateurs) : suppression d'un utilisateur
- Groups (Groupes) : suppression d'un groupe
- Roles (Rôles) : suppression d'un rôle
- Policies (Politiques) : suppression d'une politique gérée (détache aussi la politique des identités)

Utilisation des informations avant de modifier des politiques IAM

Vous pouvez passer en revue les dernières informations consultées d'une identité IAM (utilisateur, groupe ou rôle), ou d'une politique IAM avant de modifier une politique qui affecte la ressource. Ceci est important, car vous ne souhaitez pas supprimer l'accès pour une personne qui l'utilise.

Par exemple, Arnav Desai est développeur et AWS administrateur pour Example Corp. Lorsque son équipe a commencé à utiliser AWS, elle a accordé à tous les développeurs un accès utilisateur avancé qui leur a permis un accès complet à tous les services, à l'exception de l'IAM et des Organizations. Comme première étape vers l'octroi du moindre privilège, Arnav souhaite utiliser le AWS CLI pour revoir les politiques gérées dans son compte.

Pour ce faire, Arnav recense d'abord dans son compte les politiques d'autorisation gérées par le client et qui sont attachées à une identité, à l'aide de la commande suivante :

```
aws iam list-policies --scope Local --only-attached --policy-usage-filter
 PermissionsPolicy
```

À partir de la réponse, il enregistre l'ARN de chaque politique. Arnav génère ensuite un rapport sur les dernières informations consultées pour chaque politique à l'aide de la commande suivante.

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Depuis cette réponse, il capture l'ID du rapport généré depuis le champ `JobId`. Arnav interroge ensuite la commande suivante jusqu'à ce que le champ `JobStatus` renvoie la valeur `COMPLETED` ou `FAILED`. Si la tâche a échoué, il capture l'erreur.

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

Lorsque la tâche a le statut `COMPLETED`, Arnav analyse le contenu du tableau `ServicesLastAccessed` au format JSON.

```
 "ServicesLastAccessed": [
        {
            "TotalAuthenticatedEntities": 1,
            "LastAuthenticated": 2018-11-01T21:24:33.222Z,
            "ServiceNamespace": "dynamodb",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/IAMExampleUser",
            "ServiceName": "Amazon DynamoDB"
        },

        {
            "TotalAuthenticatedEntities": 0,
            "ServiceNamespace": "ec2",
            "ServiceName": "Amazon EC2"
        },

        {
            "TotalAuthenticatedEntities": 3,
            "LastAuthenticated": 2018-08-25T15:29:51.156Z,
            "ServiceNamespace": "s3",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:role/IAMExampleRole",
            "ServiceName": "Amazon S3"
        }
```

```
    ]
```

À partir de ces informations, Arnav apprend que la politique `ExamplePolicy1` autorise l'accès à trois services : Amazon DynamoDB, Amazon S3 et Amazon EC2. L'utilisateur IAM nommé `IAMExampleUser` a récemment tenté d'accéder à DynamoDB le 1er novembre, et une personne a utilisé le rôle `IAMExampleRole` pour tenter d'accéder à Amazon S3 le 25 août. Il existe également deux autres entités qui ont tenté d'accéder à Amazon S3 au cours de cette année. Cependant, personne n'a tenté d'accéder à Amazon EC2 au cours de l'année écoulée.

Cela signifie qu'Arnav peut supprimer en toute sécurité les actions Amazon EC2 de la politique. Arnav souhaite vérifier le document JSON de la politique. Tout d'abord, il doit déterminer le numéro de version de la politique à l'aide de la commande suivante.

```
aws iam list-policy-versions --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Dans la réponse, Arnav recueille le numéro de version actuelle par défaut depuis le tableau `Versions`. Il utilise ensuite le numéro de version (`v2`) pour demander le document de politique JSON à l'aide de la commande suivante.

```
aws iam get-policy-version --policy-arn arn:aws:iam::123456789012:policy/ExamplePolicy1
 --version-id v2
```

Arnav stocke le document de politique JSON renvoyé dans le champ `Document` du tableau `PolicyVersion`. Dans le document de politique, Arnav recherche les actions dans l'espace de noms `ec2`. S'il ne reste pas d'actions d'autres espaces de noms dans la politique, il détache ensuite la politique des identités affectées (utilisateurs, groupes et rôles). Il supprime ensuite la politique. Dans ce cas, la politique inclut les services Amazon DynamoDB et Amazon S3. Par conséquent, Arnav supprime les actions Amazon EC2 du document et enregistre ses modifications. Il utilise ensuite la commande suivante pour mettre à jour la politique à l'aide de la nouvelle version du document et pour définir cette version en tant que version de politique par défaut.

```
aws iam create-policy-version --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1 --policy-document file://UpdatedPolicy.json --set-as-default
```

La politique `ExamplePolicy1` est maintenant mise à jour pour supprimer l'accès au service Amazon EC2 superflu.

Autres scénarios IAM

Les informations sur la date à laquelle une ressource IAM (utilisateur, groupe, rôle ou politique) a tenté pour la dernière fois d'accéder à un service peut vous aider lorsque vous exécutez l'une des tâches suivantes :

- Policies (Politiques) : modification d'une politique en ligne ou gérée par un client pour supprimer les autorisations
- Policies (Politiques) : conversion d'une politique en ligne en une politique gérée, puis suppression
- Policies (Politiques) : ajout d'un refus explicite à une politique existante
- Policies (Politiques) : détachement d'une politique gérée d'une identité (utilisateur, groupe ou rôle)
- Entities (Entités) : définissez une limite d'autorisations pour contrôler les autorisations maximales qu'une entité (utilisateur ou rôle) peut avoir
- Groups (Groupes) : suppression d'utilisateurs d'un groupe

Utilisation des informations pour affiner les autorisations d'une unité d'organisation

Vous pouvez utiliser les dernières informations consultées pour affiner les autorisations d'une unité organisationnelle (UO) dans AWS Organizations.

Par exemple, John Stiles est AWS Organizations administrateur. Il est chargé de s'assurer que les membres de l'entreprise Comptes AWS ne disposent pas d'autorisations excessives. Dans le cadre d'un contrôle de sécurité périodique, il passe en revue les autorisations de son organisation. Son `Development` unité d'organisation contient des comptes qui sont souvent utilisés pour tester de nouveaux AWS services. John décide d'examiner périodiquement le rapport concernant les services qui n'ont pas été consultés dans plus de 180 jours. Ensuite, il supprime des autorisations d'accès à ces services pour les membres de l'unité organisationnelle.

John se connecte à la console IAM avec les informations d'identification de son compte de gestion. Dans la console IAM, il localise les données Organizations de l'unité organisationnelle `Development`. Il examine le tableau des rapports d'accès aux services et constate que deux AWS services n'ont pas été consultés depuis plus de 180 jours que sa période préférée de 180 jours. Il se souvient avoir ajouté des autorisations permettant aux équipes de développement d'accéder à Amazon Lex et AWS Database Migration Service. John contacte les équipes de développement et vérifie qu'elles n'ont plus besoin de tester ces services.

John est maintenant prêt à agir sur les dernières informations consultées. Il choisit Edit in (Faire des modifications dans) AWS Organizations et le système lui rappelle que la stratégie de contrôle de

service est attachée à plusieurs entités. Il choisit Continue (Continuer). Dans AWS Organizations, il passe en revue les cibles pour savoir à quelles Organisations le SCP est rattaché. Toutes les entités se trouvent dans l'unité organisationnelle `Development`.

John décide de refuser l'accès à l'Amazon Lex et de AWS Database Migration Service prendre des mesures au sein du `NewServiceTest` SCP. Cette action supprime l'accès inutile aux services.

Services et actions concernant les dernières informations consultées relatives à une action IAM

Le tableau suivant répertorie les AWS services pour lesquels les informations du dernier accès à l'action IAM sont affichées. Pour obtenir la liste des actions de chaque service, consultez la section Actions, ressources et clés de condition pour les AWS services dans la référence d'autorisation de service.

Service	Préfixe de service
AWS Identity and Access Management Analyseur d'accès	access-analyzer
AWS Account Management	compte
AWS Certificate Manager	acm
Amazon Managed Workflows for Apache Airflow	airflow
AWS Amplify	amplify
AWS Amplify Générateur d'interface utilisateur	amplifyuibuilder
Amazon AppIntegrations	app-integrations
AWS AppConfig	appconfig
Amazon AppFlow	appflow
AWS Profileur des coûts d'application	application-cost-profiler
Informations sur les CloudWatch applications Amazon	applicationinsights
AWS App Mesh	appmesh

Service	Préfixe de service
Amazon AppStream 2.0	appstream
AWS AppSync	appsync
Amazon Managed Service for Prometheus	aps
Amazon Athena	athena
AWS Audit Manager	auditmanager
AWS Auto Scaling	scalabilité automatique
AWS Marketplace	aws-marketplace
AWS Backup	sauvegarde
AWS Batch	lot
Amazon Braket	braket
AWS Budgets	Budgets
AWS Cloud9	Cloud9
AWS CloudFormation	cloudformation
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm
Amazon CloudSearch	cloudsearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch
AWS CodeArtifact	codeartifact
AWS CodeDeploy	codedeploy

Service	Préfixe de service
Amazon CodeGuru Profiler	codeguru-profiler
CodeGuru Réviseur Amazon	codeguru-reviewer
AWS CodePipeline	codepipeline
AWS CodeStar	codestar
AWS CodeStar Notifications	codestar-notifications
Amazon Cognito Identity	cognito-identity
Groupes d'utilisateurs Amazon Cognito	cognito-idp
Amazon Cognito Sync	cognito-sync
Amazon Comprehend Medical	comprehendmedical
AWS Compute Optimizer	compute-optimizer
AWS Config	config
Amazon Connect	connect
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	dataexchange
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax
AWS Device Farm	devicefarm
Amazon DevOps Guru	devops-guru

Service	Préfixe de service
AWS Direct Connect	directconnect
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms
Clusters élastiques Amazon DocumentDB	docdb-elastic
AWS Directory Service	ds
Amazon DynamoDB	dynamodb
Amazon Elastic Block Store	ebs
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Public	ecr-public
Amazon Elastic Container Service	ecs
Amazon Elastic Kubernetes Service	eks
Amazon Elastic Inference	elastic-inference
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	elasticbeanstalk
Amazon Elastic File System	elasticfilesystem
Elastic Load Balancing	elasticloadbalancing
Amazon Elastic Transcoder	elastictranscoder
Amazon EMR sur EKS (conteneurs EMR)	emr-containers
Amazon EMR sans serveur	emr-serverless

Service	Préfixe de service
Amazon OpenSearch Service	es
Amazon EventBridge	événements
Amazon, CloudWatch évidemment	evidently
Amazon FinSpace	finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	frauddetector
Amazon FSx	fsx
Amazon GameLift	gamelift
Amazon Location Service	geo
Amazon S3 Glacier	glacier
Amazon Managed Grafana	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	groundstation
Amazon GuardDuty	guardduty
AWS HealthLake	healthlake
Amazon Honeycode	honeycode
AWS Identity and Access Management	iam
AWS Boutique d'identités	identitystore

Service	Préfixe de service
EC2 Image Builder	imagebuilder
Amazon Inspector Classic	inspector
Amazon Inspector	inspector2
AWS IoT	iot
AWS IoT Analytics	iotanalytics
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	iottwinmaker
AWS IoT Wireless	iotwireless
Amazon Interactive Video Service	ivs
Service de vidéo interactive Amazon Chat	ivschat
Streaming géré par Amazon pour Apache Kafka	kafka
Amazon Managed Streaming for Kafka Connect	kafkaconnect
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalytics
AWS Key Management Service	kms
AWS Lambda	lambda

Service	Préfixe de service
Amazon Lex	lex
AWS License Manager Gestionnaire des abonnements Linux	license-manager-linux-subscriptions
Amazon Lightsail	lightsail
Amazon CloudWatch Logs	journaux
Amazon Lookout for Equipment	lookoutequipment
Amazon Lookout for Metrics	lookoutmetrics
Amazon Lookout for Vision	lookoutvision
AWS Mainframe Modernization	m2
Amazon Managed Blockchain	managedblockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaStore	mediastore
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB for Redis	memorydb
AWS Application Migration Service	mgn
AWS Migration Hub	mgh
AWS Migration Hub Strategy Recommendations	migrationhub-strategy
Amazon Pinpoint	mobiletargeting

Service	Préfixe de service
Amazon MQ	mq
AWS Network Manager	networkmanager
Amazon Nimble Studio	nimble
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	outposts
AWS Organizations	organisations
AWS Panorama	panorama
AWS Performance Insights	pi
Amazon EventBridge Pipes	pipes
Amazon Polly	polly
Amazon Connect Customer Profiles	profile
Amazon QLDB	qldb
AWS Resource Access Manager	ram
AWS Corbeille	rbin
Amazon Relational Database Service	rds
Amazon Redshift	redshift
API de données Amazon Redshift	redshift-data
AWS Migration Hub Refactor Spaces	refactor-spaces

Service	Préfixe de service
Amazon Rekognition	rekognition
AWS Resilience Hub	resiliencehub
Explorateur de ressources AWS	resource-explorer-2
AWS Resource Groups	resource-groups
AWS RoboMaker	robomaker
AWS Identity and Access Management Des rôles partout	rolesanywhere
Amazon Route 53	route53
Amazon Route 53 Recovery Controls	route53-recovery-control-config
Amazon Route 53 Recovery Readiness	route53-recovery-readiness
Amazon Route 53 Resolver	route53resolver
AWS CloudWatchRHUM	rum
Amazon Simple Storage Service	s3
Amazon S3 on Outposts	s3-outposts
Fonctionnalités SageMaker géospatiales d'Amazon	sagemaker-geospatial
Savings Plans	savingsplans
EventBridgeSchémas Amazon	schemas
Amazon SimpleDB	sdb
AWS Secrets Manager	secretsmanager

Service	Préfixe de service
AWS Security Hub	securityhub
Amazon Security Lake	securitylake
AWS Serverless Application Repository	serverlessrepo
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Amazon Simple Email Service	ses
AWS Shield	shield
AWS Signer	signer
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Service de messages SMS et vocaux Amazon Pinpoint	sms-voice
AWS Snowball	snowball
Amazon Simple Queue Service	sqs
AWS Systems Manager	ssm
AWS Systems Manager Incident Manager	ssm-incidents
Gestionnaire de systèmes AWS pour SAP	ssm-sap
AWS Step Functions	states
AWS Security Token Service	sts
Amazon Simple Workflow Service	swf

Service	Préfixe de service
Amazon CloudWatch Synthetics	synthetics
AWS Resource Groups Tagging API	balise
Amazon Textract	textract
Amazon Timestream	timestream
AWS Générateur de réseaux de télécommunications	tnb
Amazon Transcribe	transcribe
AWS Transfer Family	transfert
Amazon Translate	translate
Amazon Connect Voice ID	voiceid
Amazon VPC Lattice	vpc-lattice
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Amazon Connect Wisdom	wisdom
Amazon WorkLink	worklink
Amazon WorkSpaces	espaces de travail
AWS X-Ray	xray

Actions concernant les dernières informations consultées relatives à une action

Le tableau suivant répertorie les actions pour lesquelles les dernières informations consultées relatives à une action sont disponibles.

Préfixe de service	Actions
access-analyzer	access-analyzer : Règle ApplyArchive
	analyseur d'accès : Génération CancelPolicy
	analyseur d'accès : CheckAccess NotGranted
	analyseur d'accès : CheckNo NewAccess
	access-analyzer : Aperçu CreateAccess
	analyseur d'accès : CreateAnalyzer
	access-analyzer : Règle CreateArchive
	analyseur d'accès : DeleteAnalyzer
	access-analyzer : Règle DeleteArchive
	access-analyzer : Aperçu GetAccess
	access-analyzer : Ressource GetAnalyzed
	analyseur d'accès : GetAnalyzer
	access-analyzer : Règle GetArchive
	analyseur d'accès : GetFinding
	access-analyzer : Politique GetGenerated
	analyseur d'accès : ListAccess PreviewFindings
	access-analyzer : aperçus ListAccess
	access-analyzer : Ressources ListAnalyzed
	analyseur d'accès : ListAnalyzers
	access-analyzer : Règles ListArchive
	analyseur d'accès : ListFindings

Préfixe de service	Actions
	analyseur d'accès : Generations ListPolicy analyseur d'accès : Génération StartPolicy analyseur d'accès : Scan StartResource access-analyzer : Règle UpdateArchive analyseur d'accès : UpdateFindings analyseur d'accès : ValidatePolicy
compte	compte : DeleteAlternate Contact compte : DisableRegion compte : EnableRegion compte : GetAlternate Contact compte : GetContact Informations compte : GetRegion OptStatus compte : ListRegions compte : PutAlternate Contact compte : PutContact Informations

Préfixe de service	Actions
acm	ACM : DeleteCertificate ACM : DescribeCertificate ACM : ExportCertificate acm : Configuration GetAccount ACM : GetCertificate ACM : ImportCertificate ACM : ListCertificates acm : Configuration PutAccount ACM : RenewCertificate ACM : RequestCertificate acm : Courrier électronique ResendValidation ACM : Options UpdateCertificate
airflow	flux d'air : CreateCli jeton débit d'air : CreateEnvironment débit d'air : CreateWeb LoginToken débit d'air : DeleteEnvironment débit d'air : GetEnvironment débit d'air : ListEnvironments débit d'air : PublishMetrics débit d'air : UpdateEnvironment

Préfixe de service	Actions
amplify	amplifier : CreateApp
	amplifier : Environnement CreateBackend
	amplifier : CreateBranch
	amplifier : CreateDeployment
	amplifier : Association CreateDomain
	amplifier : Hook CreateWeb
	amplifier : DeleteApp
	amplifier : Environnement DeleteBackend
	amplifier : DeleteBranch
	amplifier : Association DeleteDomain
	amplifier : DeleteJob
	amplifier : Hook DeleteWeb
	amplifier : Logs GenerateAccess
	amplifier : GetApp
	amplifier : URL GetArtifact
	amplifier : Environnement GetBackend
	amplifier : GetBranch
	amplifier : Association GetDomain
	amplifier : GetJob
	amplifier : Hook GetWeb
	amplifier : ListApps

Préfixe de service	Actions
	amplifier : ListArtifacts
	amplifier : Environnements ListBackend
	amplifier : ListBranches
	amplifier : Associations ListDomain
	amplifier : ListJobs
	amplifier : Hooks ListWeb
	amplifier : StartDeployment
	amplifier : StartJob
	amplifier : StopJob
	amplifier : UpdateApp
	amplifier : UpdateBranch
	amplifier : Association UpdateDomain
	amplifier : Hook UpdateWeb

Préfixe de service	Actions
amplifyuibuilder	Amplify UI Builder : CreateComponent Amplify UI Builder : CreateForm Amplify UI Builder : CreateTheme Amplify UI Builder : DeleteComponent Amplify UI Builder : DeleteForm Amplify UI Builder : DeleteTheme Amplify UI Builder : ExportComponents Amplify UI Builder : ExportThemes amplifyuibuilder : Job GetCodegen amplifyuibuilder : Offres d'emploi ListCodegen Amplify UI Builder : ListComponents Amplify UI Builder : ListForms Amplify UI Builder : ListThemes amplifyuibuilder : Drapeau ResetMetadata amplifyuibuilder : Job StartCodegen Amplify UI Builder : UpdateComponent Amplify UI Builder : UpdateForm Amplify UI Builder : UpdateTheme

Préfixe de service	Actions
app-integrations	intégrations d'applications : CreateApplication
	intégrations d'applications : Intégration CreateData
	intégrations d'applications : Intégration CreateEvent
	intégrations d'applications : DeleteApplication
	intégrations d'applications : Intégration DeleteData
	intégrations d'applications : Intégration DeleteEvent
	intégrations d'applications : GetApplication
	intégrations d'applications : Intégration GetData
	intégrations d'applications : Intégration GetEvent
	intégrations d'applications : Associations ListApplication
	intégrations d'applications : ListApplications
	intégrations d'applications : ListData IntegrationAssociations
	intégrations d'applications : Intégrations ListData
	intégrations d'applications : ListEvent IntegrationAssociations
	intégrations d'applications : Intégrations ListEvent
	intégrations d'applications : UpdateApplication
	intégrations d'applications : Intégration UpdateData
	intégrations d'applications : Intégration UpdateEvent

Préfixe de service	Actions
appconfig	configuration de l'application : CreateApplication appconfig : Profil CreateConfiguration appconfig : Stratégie CreateDeployment configuration de l'application : CreateEnvironment configuration de l'application : CreateExtension appconfig : Association CreateExtension configuration de l'application : CreateHosted ConfigurationVersion configuration de l'application : DeleteApplication appconfig : Profil DeleteConfiguration appconfig : Stratégie DeleteDeployment configuration de l'application : DeleteEnvironment configuration de l'application : DeleteExtension appconfig : Association DeleteExtension configuration de l'application : DeleteHosted ConfigurationVersion configuration de l'application : GetApplication configuration de l'application : GetConfiguration appconfig : Profil GetConfiguration configuration de l'application : GetDeployment appconfig : Stratégie GetDeployment configuration de l'application : GetEnvironment configuration de l'application : GetExtension

Préfixe de service	Actions
	appconfig : Association GetExtension configuration de l'application : GetHosted ConfigurationVersion configuration de l'application : ListApplications appconfig : Profils ListConfiguration configuration de l'application : ListDeployments appconfig : Stratégies ListDeployment configuration de l'application : ListEnvironments appconfig : Associations ListExtension configuration de l'application : ListExtensions configuration de l'application : ListHosted ConfigurationVersions configuration de l'application : StartDeployment configuration de l'application : StopDeployment configuration de l'application : UpdateApplication appconfig : Profil UpdateConfiguration appconfig : Stratégie UpdateDeployment configuration de l'application : UpdateEnvironment configuration de l'application : UpdateExtension appconfig : Association UpdateExtension configuration de l'application : ValidateConfiguration

Préfixe de service	Actions
appflow	appflow : Exécutions CancelFlow
	appflow : Profil CreateConnector
	flux d'applications : CreateFlow
	appflow : Profil DeleteConnector
	flux d'applications : DeleteFlow
	flux d'applications : DescribeConnector
	appflow : Entité DescribeConnector
	appflow : Profils DescribeConnector
	flux d'applications : DescribeConnectors
	flux d'applications : DescribeFlow
	flux d'applications : DescribeFlow ExecutionRecords
	appflow : Entités ListConnector
	flux d'applications : ListConnectors
	flux d'applications : ListFlows
	flux d'applications : RegisterConnector
	flux d'applications : ResetConnector MetadataCache
	flux d'applications : StartFlow
	flux d'applications : StopFlow
	appflow : Connecteur UnRegister
	appflow : Profil UpdateConnector
	appflow : Inscription UpdateConnector

Préfixe de service	Actions
	flux d'applications : UpdateFlow
application-cost-profiler	profileur de coûts d'application : définition DeleteReport profileur de coûts d'application : définition GetReport application-cost profiler : utilisation ImportApplication profileur de coûts d'application : définitions ListReport profileur de coûts d'application : définition PutReport profileur de coûts d'application : définition UpdateReport

Préfixe de service	Actions
applicationinsights	informations sur les applications : AddWorkload
	informations sur les applications : CreateApplication
	informations sur les applications : CreateComponent
	Informations sur les applications : modèle CreateLog
	informations sur les applications : DeleteApplication
	informations sur les applications : DeleteComponent
	Informations sur les applications : modèle DeleteLog
	informations sur les applications : DescribeApplication
	informations sur les applications : DescribeComponent
	Informations sur les applications : Configuration DescribeC omponent
	informations sur les applications : DescribeComponent Configura tionRecommendation
	Informations sur les applications : modèle DescribeLog
	informations sur les applications : DescribeObservation
	informations sur les applications : DescribeProblem
	informations sur les applications : DescribeProblem observations
	informations sur les applications : DescribeWorkload
	informations sur les applications : ListApplications
	informations sur les applications : ListComponents
	applicationinsights : Historique ListConfiguration
	Informations sur les applications : modèles ListLog

Préfixe de service	Actions
	informations sur les applications : ListLog PatternSets
	informations sur les applications : ListProblems
	informations sur les applications : ListWorkloads
	informations sur les applications : RemoveWorkload
	informations sur les applications : UpdateApplication
	informations sur les applications : UpdateComponent
	Informations sur les applications : Configuration UpdateComponent
	Informations sur les applications : modèle UpdateLog
	informations sur les applications : UpdateWorkload

Préfixe de service	Actions
appmesh	appmesh : Route CreateGateway maillage d'applications : CreateMesh maillage d'applications : CreateRoute appmesh : Passerelle CreateVirtual appmesh : Nœud CreateVirtual appmesh : Routeur CreateVirtual appmesh : Service CreateVirtual appmesh : Route DeleteGateway maillage d'applications : DeleteMesh maillage d'applications : DeleteRoute appmesh : Passerelle DeleteVirtual appmesh : Nœud DeleteVirtual appmesh : Routeur DeleteVirtual appmesh : Service DeleteVirtual appmesh : Route DescribeGateway maillage d'applications : DescribeMesh maillage d'applications : DescribeRoute appmesh : Passerelle DescribeVirtual appmesh : Nœud DescribeVirtual appmesh : Routeur DescribeVirtual appmesh : Service DescribeVirtual

Préfixe de service	Actions
	appmesh : Itinéraires ListGateway maillage d'applications : ListMeshes maillage d'applications : ListRoutes appmesh : Passerelles ListVirtual appmesh : Nœuds ListVirtual appmesh : Routeurs ListVirtual appmesh : Prestations ListVirtual appmesh : Ressources StreamAggregated appmesh : Route UpdateGateway maillage d'applications : UpdateMesh maillage d'applications : UpdateRoute appmesh : Passerelle UpdateVirtual appmesh : Nœud UpdateVirtual appmesh : Routeur UpdateVirtual appmesh : Service UpdateVirtual

Préfixe de service	Actions
appstream	appstream : AssociateApp BlockBuilder AppBlock
	appstream : Flotte AssociateApplication
	appstream : AssociateApplication ToEntitlement
	appstream : AssociateFleet
	appstream : BatchAssociate UserStack
	appstream : BatchDisassociate UserStack
	appstream : CopyImage
	appstream : Bloquer CreateApp
	appstream : CreateApp BlockBuilder
	appstream : URL de diffusion CreateApp BlockBuilder
	appstream : CreateApplication
	appstream : Config CreateDirectory
	appstream : CreateEntitlement
	appstream : CreateFleet
	appstream : Générateur CreateImage
	appstream : URL CreateImage BuilderStreaming
	appstream : CreateStack
	appstream : URL CreateStreaming
	appstream : Image CreateUpdated
	appstream : CreateUsage ReportSubscription
	appstream : CreateUser

Préfixe de service	Actions
	appstream : Bloquer DeleteApp
	appstream : DeleteApp BlockBuilder
	appstream : DeleteApplication
	appstream : Config DeleteDirectory
	appstream : DeleteEntitlement
	appstream : DeleteFleet
	appstream : DeleteImage
	appstream : Générateur DeleteImage
	appstream : Autorisations DeleteImage
	appstream : DeleteStack
	appstream : DeleteUsage ReportSubscription
	appstream : DeleteUser
	appstream : Associations DescribeApp BlockBuilder AppBlock
	appstream : DescribeApp BlockBuilders
	appstream : Blocs DescribeApp
	appstream : DescribeApplication FleetAssociations
	appstream : DescribeApplications
	appstream : Configurations DescribeDirectory
	appstream : DescribeEntitlements
	appstream : DescribeFleets
	appstream : Constructeurs DescribeImage

Préfixe de service	Actions
	appstream : Autorisations DescribeImage
	appstream : DescribeImages
	appstream : DescribeSessions
	appstream : DescribeStacks
	appstream : DescribeUsage ReportSubscriptions
	appstream : DescribeUsers
	appstream : DescribeUser StackAssociations
	appstream : DisableUser
	appstream : DisassociateApp BlockBuilder AppBlock
	appstream : Flotte DisassociateApplication
	appstream : DisassociateApplication FromEntitlement
	appstream : DisassociateFleet
	appstream : EnableUser
	appstream : ExpireSession
	appstream : Flottes ListAssociated
	appstream : Stacks ListAssociated
	appstream : Applications ListEntitled
	appstream : StartApp BlockBuilder
	appstream : StartFleet
	appstream : Générateur StartImage
	appstream : StopApp BlockBuilder

Préfixe de service	Actions
	appstream : StopFleet
	appstream : Générateur StopImage
	appstream : UpdateApp BlockBuilder
	appstream : UpdateApplication
	appstream : Config UpdateDirectory
	appstream : UpdateEntitlement
	appstream : UpdateFleet
	appstream : Autorisations UpdateImage
	appstream : UpdateStack

Préfixe de service	Actions
appsync	synchronisation des applications : AssociateApi synchronisation des applications : AssociateMerged GraphqlApi synchronisation des applications : AssociateSource GraphqlApi appsync : cache CreateApi appsync : clé CreateApi appsync : Source CreateData appsync : Nom CreateDomain synchronisation des applications : CreateFunction appsync : API CreateGraphql synchronisation des applications : CreateResolver synchronisation des applications : CreateType appsync : cache DeleteApi appsync : clé DeleteApi appsync : Source DeleteData appsync : Nom DeleteDomain synchronisation des applications : DeleteFunction appsync : API DeleteGraphql synchronisation des applications : DeleteResolver synchronisation des applications : DeleteType synchronisation des applications : DisassociateApi synchronisation des applications : DisassociateMerged GraphqlApi

Préfixe de service	Actions
	synchronisation des applications : DisassociateSource GraphqlApi synchronisation des applications : EvaluateCode appsync : Modèle EvaluateMapping appsync : cache FlushApi appsync : Association GetApi appsync : cache GetApi appsync : Source GetData synchronisation des applications : GetData SourceIntrospection appsync : Nom GetDomain synchronisation des applications : GetFunction appsync : API GetGraphql appsync : GetGraphql ApiEnvironment Variables appsync : Schéma GetIntrospection synchronisation des applications : GetResolver synchronisation des applications : GetSchema CreationStatus synchronisation des applications : GetSource ApiAssociation synchronisation des applications : GetType appsync : Clés ListApi appsync : Sources ListData appsync : Noms ListDomain synchronisation des applications : ListFunctions

Préfixe de service	Actions
	appsync : API ListGraphql synchronisation des applications : ListResolvers synchronisation des applications : ListResolvers ByFunction synchronisation des applications : ListSource ApiAssociations synchronisation des applications : ListTypes synchronisation des applications : ListTypes ByAssociation appsync : PutGraphql ApiEnvironment Variables synchronisation des applications : StartData SourceIntrospection appsync : Création StartSchema appsync : Fusionner StartSchema appsync : cache UpdateApi appsync : clé UpdateApi appsync : Source UpdateData appsync : Nom UpdateDomain synchronisation des applications : UpdateFunction appsync : API UpdateGraphql synchronisation des applications : UpdateResolver synchronisation des applications : UpdateSource ApiAssociation synchronisation des applications : UpdateType

Préfixe de service	Actions
aps	Cartes : CreateAlert ManagerDefinition
	Cartes : CreateLogging Configuration
	Cartes : CreateRule GroupsNamespace
	Cartes : CreateScraper
	Cartes : CreateWorkspace
	Cartes : DeleteAlert ManagerDefinition
	Cartes : DeleteLogging Configuration
	Cartes : DeleteRule GroupsNamespace
	Cartes : DeleteScraper
	Cartes : DeleteWorkspace
	Cartes : DescribeAlert ManagerDefinition
	Cartes : DescribeLogging Configuration
	Cartes : DescribeRule GroupsNamespace
	Cartes : DescribeScraper
	Cartes : DescribeWorkspace
	Cartes : GetDefault ScraperConfiguration
	Cartes : ListRule GroupsNamespaces
	Cartes : ListScrapers
	Cartes : ListWorkspaces
	Cartes : PutAlert ManagerDefinition
	Cartes : PutRule GroupsNamespace

Préfixe de service	Actions
	Cartes : UpdateLogging Configuration Cartes : UpdateWorkspace Alias

Préfixe de service	Actions
athena	Athéna : BatchGet NamedQuery Athéna : BatchGet PreparedStatement Athéna : BatchGet QueryExecution athena : Réservation CancelCapacity athena : Réservation CreateCapacity athena : Catalogue CreateData athena : Requête CreateNamed Athéna : CreateNotebook athena : Déclaration CreatePrepared Athéna : CreatePresigned NotebookUrl athena : Groupe CreateWork athena : Réservation DeleteCapacity athena : Catalogue DeleteData athena : Requête DeleteNamed Athéna : DeleteNotebook athena : Déclaration DeletePrepared athena : Groupe DeleteWork Athéna : ExportNotebook athena : Exécution GetCalculation Athéna : GetCalculation ExecutionCode Athéna : GetCalculation ExecutionStatus

Préfixe de service	Actions
	Athéna : GetCapacity AssignmentConfiguration athena : Réservation GetCapacity Athéna : GetDatabase athena : Catalogue GetData athena : Requête GetNamed athena : Métadonnées GetNotebook athena : Déclaration GetPrepared athena : Exécution GetQuery Athena : Résultats GetQuery Athéna : GetQuery ResultsStream Athéna : GetQuery RuntimeStatistics Athéna : GetSession athena : Statut GetSession athena : Métadonnées GetTable athena : Groupe GetWork Athéna : ImportNotebook athena : DPUSizes ListApplication Athéna : Exécutions ListCalculation athena : Réservations ListCapacity Athéna : ListDatabases athena : Catalogues ListData

Préfixe de service	Actions
	Athéna : Versions ListEngine Athéna : ListExecutors athena : Requêtes ListNamed athena : Métadonnées ListNotebook Athéna : Sessions ListNotebook athena : Déclarations ListPrepared Athéna : Exécutions ListQuery Athéna : ListSessions athena : Métadonnées ListTable athena : Groupes ListWork Athéna : PutCapacity AssignmentConfiguration athena : Exécution StartCalculation athena : Exécution StartQuery Athéna : StartSession athena : Exécution StopCalculation athena : Exécution StopQuery Athéna : TerminateSession athena : Réservation UpdateCapacity athena : Catalogue UpdateData athena : Requête UpdateNamed Athéna : UpdateNotebook

Préfixe de service	Actions
	athena : Métadonnées UpdateNotebook athena : Déclaration UpdatePrepared athena : Groupe UpdateWork

Préfixe de service	Actions
auditmanager	auditmanager : Dossier AssociateAssessment ReportEvidence
	auditmanager : Preuves BatchAssociate AssessmentReport
	auditmanager : Évaluation BatchCreate DelegationBy
	auditmanager : Évaluation BatchDelete DelegationBy
	auditmanager : Preuves BatchDisassociate AssessmentReport
	responsable de l'audit : BatchImport EvidenceTo Assessmen tControl
	responsable de l'audit : CreateAssessment
	auditmanager : Framework CreateAssessment
	auditmanager : Rapport CreateAssessment
	responsable de l'audit : CreateControl
	responsable de l'audit : DeleteAssessment
	auditmanager : Framework DeleteAssessment
	responsable de l'audit : DeleteAssessment FrameworkShare
	auditmanager : Rapport DeleteAssessment
	responsable de l'audit : DeleteControl
	responsable de l'audit : DeregisterAccount
	responsable de l'audit : DeregisterOrganization AdminAccount
	auditmanager : Dossier DisassociateAssessment ReportEvidence
	auditmanager : État GetAccount
	responsable de l'audit : GetAssessment

Préfixe de service	Actions
	auditmanager : Framework GetAssessment
	responsable de l'audit : GetAssessment ReportUrl
	auditmanager : GetChange Journaux
	responsable de l'audit : GetControl
	responsable de l'audit : GetDelegations
	responsable de l'audit : GetEvidence
	auditmanager : Dossier GetEvidence ByEvidence
	auditmanager : URL GetEvidence FileUpload
	auditmanager : Dossier GetEvidence
	auditmanager : Évaluation GetEvidence FoldersBy
	responsable de l'audit : GetEvidence FoldersBy AssessmentControl
	responsable de l'audit : GetInsights
	responsable de l'audit : GetInsights ByAssessment
	responsable de l'audit : GetOrganization AdminAccount
	responsable de l'audit : GetServices InScope
	responsable de l'audit : GetSettings
	auditmanager : Domaine ListAssessment ControlInsights ByControl
	auditmanager : Frameworks ListAssessment
	auditmanager : Demandes ListAssessment FrameworkShare
	auditmanager : Rapports ListAssessment
	responsable de l'audit : ListAssessments

Préfixe de service	Actions
	responsable de l'audit : ListControl DomainInsights
	responsable de l'audit : ListControl DomainInsights ByAssessment
	responsable de l'audit : ListControl InsightsBy ControlDomain
	responsable de l'audit : ListControls
	responsable de l'audit : Source ListKeywords ForData
	responsable de l'audit : ListNotifications
	responsable de l'audit : RegisterAccount
	responsable de l'audit : RegisterOrganization AdminAccount
	responsable de l'audit : StartAssessment FrameworkShare
	responsable de l'audit : UpdateAssessment
	auditmanager : Contrôle UpdateAssessment
	auditmanager : État UpdateAssessment ControlSet
	auditmanager : Framework UpdateAssessment
	responsable de l'audit : UpdateAssessment FrameworkShare
	auditmanager : État UpdateAssessment
	responsable de l'audit : UpdateControl
	responsable de l'audit : UpdateSettings
	responsable de l'audit : ValidateAssessment ReportIntegrity

Préfixe de service	Actions
scalabilité automatique	mise à l'échelle automatique : AttachInstances autoscaling : équilibreurs AttachLoad mise à l'échelle automatique : groupes AttachLoad BalancerTarget mise à l'échelle automatique : Sources AttachTraffic mise à l'échelle automatique : BatchDelete ScheduledAction mise à l'échelle automatique : BatchPut ScheduledUpdate GroupAction mise à l'échelle automatique : Actualiser CancelInstance mise à l'échelle automatique : Action CompleteLifecycle mise à l'échelle automatique : CreateAuto ScalingGroup mise à l'échelle automatique : Configuration CreateLaunch mise à l'échelle automatique : DeleteAuto ScalingGroup mise à l'échelle automatique : Configuration DeleteLaunch mise à l'échelle automatique : Hook DeleteLifecycle mise à l'échelle automatique : Configuration DeleteNotification mise à l'échelle automatique : DeletePolicy mise à l'échelle automatique : Action DeleteScheduled mise à l'échelle automatique : pool DeleteWarm mise à l'échelle automatique : limites DescribeAccount mise à l'échelle automatique : types DescribeAdjustment mise à l'échelle automatique : DescribeAuto ScalingGroups

Préfixe de service	Actions
	mise à l'échelle automatique : DescribeAuto ScalingInstances mise à l'échelle automatique : types DescribeAuto ScalingNotificatio n mise à l'échelle automatique : actualise DescribeInstance mise à l'échelle automatique : configurations DescribeLaunch mise à l'échelle automatique : Hooks DescribeLifecycle mise à l'échelle automatique : DescribeLifecycle HookTypes autoscaling : équilibreurs DescribeLoad mise à l'échelle automatique : groupes DescribeLoad BalancerT arget mise à l'échelle automatique : DescribeMetric CollectionTypes mise à l'échelle automatique : configurations DescribeNotification mise à l'échelle automatique : DescribePolicies autoscaling : Activités DescribeScaling mise à l'échelle automatique : DescribeScaling ProcessTypes mise à l'échelle automatique : actions DescribeScheduled mise à l'échelle automatique : DescribeTermination PolicyTypes mise à l'échelle automatique : Sources DescribeTraffic mise à l'échelle automatique : pool DescribeWarm mise à l'échelle automatique : DetachInstances autoscaling : équilibreurs DetachLoad mise à l'échelle automatique : groupes DetachLoad BalancerTarget

Préfixe de service	Actions
	mise à l'échelle automatique : Sources DetachTraffic mise à l'échelle automatique : collection DisableMetrics mise à l'échelle automatique : collection EnableMetrics mise à l'échelle automatique : EnterStandby mise à l'échelle automatique : ExecutePolicy mise à l'échelle automatique : ExitStandby mise à l'échelle automatique : GetPredictive ScalingForecast mise à l'échelle automatique : Hook PutLifecycle mise à l'échelle automatique : Configuration PutNotification autoscaling : Politique PutScaling mise à l'échelle automatique : Action PutScheduled UpdateGroup mise à l'échelle automatique : pool PutWarm mise à l'échelle automatique : RecordLifecycle ActionHeartbeat mise à l'échelle automatique : ResumeProcesses mise à l'échelle automatique : Actualiser RollbackInstance autoscaling : Capacité SetDesired mise à l'échelle automatique : Health SetInstance mise à l'échelle automatique : protection SetInstance mise à l'échelle automatique : Actualiser StartInstance mise à l'échelle automatique : SuspendProcesses

Préfixe de service	Actions
	mise à l'échelle automatique : TerminateInstance InAuto ScalingGroup mise à l'échelle automatique : UpdateAuto ScalingGroup
aws-marketplace	AWS Marketplace : GetEntitlements

Préfixe de service	Actions
sauvegarde	sauvegarde : CancelLegal Hold sauvegarde : CreateBackup Planifier sauvegarde : CreateBackup Sélection sauvegarde : CreateBackup Vault sauvegarde : CreateFramework sauvegarde : CreateLegal Hold sauvegarde : CreateLogically AirGapped BackupVault sauvegarde : CreateReport Planifier sauvegarde : CreateRestore TestingPlan sauvegarde : CreateRestore TestingSelection sauvegarde : DeleteBackup Planifier sauvegarde : DeleteBackup Sélection sauvegarde : DeleteBackup Vault sauvegarde : DeleteBackup VaultAccess Politique sauvegarde : DeleteBackup VaultLock Configuration sauvegarde : DeleteBackup VaultNotifications sauvegarde : DeleteFramework sauvegarde : DeleteRecovery Point sauvegarde : DeleteReport Planifier sauvegarde : DeleteRestore TestingPlan sauvegarde : DeleteRestore TestingSelection

Préfixe de service	Actions
	sauvegarde : DescribeBackup Job sauvegarde : DescribeBackup Vault sauvegarde : DescribeCopy Job sauvegarde : DescribeFramework sauvegarde : DescribeGlobal Paramètres sauvegarde : DescribeProtected ressource sauvegarde : DescribeRecovery Point sauvegarde : DescribeRegion Paramètres sauvegarde : DescribeReport Job sauvegarde : DescribeReport Planifier sauvegarde : DescribeRestore Job sauvegarde : DisassociateRecovery Point sauvegarde : DisassociateRecovery PointFrom Parent sauvegarde : ExportBackup PlanTemplate sauvegarde : GetBackup Planifier sauvegarde : GetBackup PlanFrom JSON sauvegarde : GetBackup PlanFrom modèle sauvegarde : GetBackup Sélection sauvegarde : GetBackup VaultAccess Politique sauvegarde : GetBackup VaultNotifications sauvegarde : GetLegal Hold

Préfixe de service	Actions
	sauvegarde : GetRecovery PointRestore métadonnées sauvegarde : GetRestore JobMetadata sauvegarde : GetRestore TestingInferred métadonnées sauvegarde : GetRestore TestingPlan sauvegarde : GetRestore TestingSelection sauvegarde : GetSupported ResourceTypes sauvegarde : ListBackup Jobs sauvegarde : ListBackup JobSummaries sauvegarde : ListBackup Plans sauvegarde : ListBackup PlanTemplates sauvegarde : ListBackup PlanVersions sauvegarde : ListBackup Sélections sauvegarde : ListBackup coffres-forts sauvegarde : ListCopy Jobs sauvegarde : ListCopy JobSummaries sauvegarde : ListFrameworks sauvegarde : ListLegal Holds sauvegarde : ListProtected Ressources sauvegarde : ListRecovery PointsBy BackupVault sauvegarde : ListRecovery PointsBy LegalHold sauvegarde : ListRecovery PointsBy ressource

Préfixe de service	Actions
	sauvegarde : ListReport Jobs sauvegarde : ListReport Plans sauvegarde : ListRestore Jobs sauvegarde : ListRestore JobsBy ProtectedResource sauvegarde : ListRestore JobSummaries sauvegarde : ListRestore TestingPlans sauvegarde : ListRestore TestingSelections sauvegarde : PutBackup VaultAccess Politique sauvegarde : PutBackup VaultLock Configuration sauvegarde : PutBackup VaultNotifications sauvegarde : PutRestore ValidationResult sauvegarde : StartBackup Job sauvegarde : StartCopy Job sauvegarde : StartReport Job sauvegarde : StartRestore Job sauvegarde : StopBackup Job sauvegarde : UpdateBackup Planifier sauvegarde : UpdateFramework sauvegarde : UpdateGlobal Paramètres sauvegarde : UpdateRecovery PointLifecycle sauvegarde : UpdateRegion Paramètres

Préfixe de service	Actions
	sauvegarde : UpdateReport Planifier sauvegarde : UpdateRestore TestingPlan sauvegarde : UpdateRestore TestingSelection

Préfixe de service	Actions
lot	lot : CancelJob lot : CreateCompute Environnement batch : CreateJob file d'attente batch : CreateScheduling Politique lot : DeleteCompute Environnement batch : DeleteJob file d'attente batch : DeleteScheduling Politique batch : DeregisterJob définition lot : DescribeCompute Environnements lot : DescribeJob Définitions lot : files d'DescribeJobattente lot : DescribeJobs batch : DescribeScheduling Politiques lot : ListJobs batch : ListScheduling Politiques batch : RegisterJob définition lot : SubmitJob lot : TerminateJob lot : UpdateCompute Environnement batch : UpdateJob file d'attente batch : UpdateScheduling Politique

Préfixe de service	Actions
braket	support : accord AcceptUser support : fonctionnalité AccessBraket support : CancelJob support : Task CancelQuantum support : CreateJob support : Task CreateQuantum support : GetDevice support : GetJob support : Task GetQuantum support : Status GetService LinkedRole support : GetUser AgreementStatus support : SearchDevices support : SearchJobs support : Tâches SearchQuantum

Préfixe de service	Actions
Budgets	budgets : ModifyBudget
	budgets : CreateBudget Action
	budgets : ModifyBudget
	budgets : ModifyBudget
	budgets : ModifyBudget
	budgets : DeleteBudget Action
	budgets : ModifyBudget
	budgets : ModifyBudget
	budgets : ViewBudget
	budgets : DescribeBudget Action
	budgets : DescribeBudget ActionHistories
	budgets : DescribeBudget ActionsFor Compte
	budgets : DescribeBudget ActionsFor Budget
	budgets : ViewBudget
	budgets : ViewBudget
	budgets : ViewBudget
	budgets : ViewBudget
	budgets : ViewBudget
	budgets : ExecuteBudget Action
	budgets : ModifyBudget
	budgets : UpdateBudget Action

Préfixe de service	Actions
	budgets : ModifyBudget budgets : ModifyBudget
Cloud9	cloud 9 : EC2 CreateEnvironment cloud9 : Adhésion CreateEnvironment nuage 9 : DeleteEnvironment cloud9 : Adhésion DeleteEnvironment cloud9 : Adhésions DescribeEnvironment nuage 9 : DescribeEnvironments cloud9 : État DescribeEnvironment nuage 9 : ListEnvironments nuage 9 : UpdateEnvironment cloud9 : Adhésion UpdateEnvironment

Préfixe de service	Actions
cloudformation	formation des nuages : BatchDescribe TypeConfigurations
	formation du cloud : Stack CancelUpdate
	cloudformation : Annulation ContinueUpdate
	cloudformation : Définir CreateChange
	cloudformation : Modèle CreateGenerated
	formation des nuages : CreateStack
	cloudformation : instances CreateStack
	cloudformation : Définir CreateStack
	formation des nuages : DeactivateType
	cloudformation : Définir DeleteChange
	cloudformation : Modèle DeleteGenerated
	formation des nuages : DeleteStack
	cloudformation : instances DeleteStack
	cloudformation : Définir DeleteStack
	formation des nuages : DeregisterType
	cloudformation : limites DescribeAccount
	cloudformation : Définir DescribeChange
	formation des nuages : DescribeChange SetHooks
	cloudformation : Modèle DescribeGenerated
	cloudformation : Accès DescribeOrganizations
	formation des nuages : DescribePublisher

Préfixe de service	Actions
	cloudformation : Scan DescribeResource
	cloudformation : État DescribeStack DriftDetection
	cloudformation : Événements DescribeStack
	cloudformation : Instance DescribeStack
	cloudformation : Ressource DescribeStack
	formation des nuages : DescribeStack ResourceDrifts
	cloudformation : Ressources DescribeStack
	formation des nuages : DescribeStacks
	cloudformation : Définir DescribeStack
	formation des nuages : DescribeStack SetOperation
	formation des nuages : DescribeType
	cloudformation : Inscription DescribeType
	formation nuageuse : Drift DetectStack
	formation des nuages : DetectStack ResourceDrift
	formation des nuages : DetectStack SetDrift
	cloudformation : coût EstimateTemplate
	cloudformation : Définir ExecuteChange
	cloudformation : Modèle GetGenerated
	cloudformation : Politique GetStack
	formation des nuages : GetTemplate
	cloudformation : Résumé GetTemplate

Préfixe de service	Actions
	cloudformation : Définir ImportStacks ToStack
	cloudformation : Ensembles ListChange
	formation des nuages : ListExports
	cloudformation : Modèles ListGenerated
	formation des nuages : ListImports
	cloudformation : Ressources ListResource ScanRelated
	formation des nuages : ListResource ScanResources
	cloudformation : Scans ListResource
	formation de nuages : Drifts ListStack InstanceResource
	cloudformation : instances ListStack
	cloudformation : Ressources ListStack
	formation des nuages : ListStack SetAuto DeploymentTargets
	cloudformation : Résultats ListStack SetOperation
	formation des nuages : ListStack SetOperations
	cloudformation : Ensembles ListStack
	cloudformation : Inscriptions ListType
	formation des nuages : ListTypes
	cloudformation : Versions ListType
	formation des nuages : PublishType
	cloudformation : Progrès RecordHandler
	formation des nuages : RegisterPublisher

Préfixe de service	Actions
	formation des nuages : RegisterType
	formation des nuages : RollbackStack
	cloudformation : Politique SetStack
	cloudformation : Configuration SetType
	formation des nuages : SetType DefaultVersion
	formation des nuages : SignalResource
	cloudformation : Scan StartResource
	formation des nuages : StopStack SetOperation
	formation des nuages : TestType
	cloudformation : Modèle UpdateGenerated
	formation des nuages : UpdateStack
	cloudformation : instances UpdateStack
	cloudformation : Définir UpdateStack
	formation sur le cloud : protection UpdateTermination
	formation des nuages : ValidateTemplate

Préfixe de service	Actions
cloudfront	front de nuage : AssociateAlias cloudfront : Politique CreateCache front de nuage : CreateCloud FrontOrigin AccessIdentity front de nuage : CreateContinuous DeploymentPolicy cloudfront : Config CreateField LevelEncryption cloudfront : Profil CreateField LevelEncryption front de nuage : CreateFunction front de nuage : CreateInvalidation cloudfront : Groupe CreateKey front de nuage : CreateKey ValueStore cloudfront : Abonnement CreateMonitoring front de nuage : CreateOrigin AccessControl front de nuage : CreateOrigin RequestPolicy cloudfront : clé CreatePublic front de nuage : CreateRealtime LogConfig front de nuage : CreateResponse HeadersPolicy cloudfront : Politique DeleteCache front de nuage : DeleteCloud FrontOrigin AccessIdentity front de nuage : DeleteContinuous DeploymentPolicy front de nuage : DeleteDistribution cloudfront : Config DeleteField LevelEncryption

Préfixe de service	Actions
	cloudfront : Profil DeleteField LevelEncryption front de nuage : DeleteFunction cloudfront : Groupe DeleteKey front de nuage : DeleteKey ValueStore cloudfront : Abonnement DeleteMonitoring front de nuage : DeleteOrigin AccessControl front de nuage : DeleteOrigin RequestPolicy cloudfront : clé DeletePublic front de nuage : DeleteRealtime LogConfig front de nuage : DeleteResponse HeadersPolicy cloudfront : Distribution DeleteStreaming front de nuage : DescribeFunction front de nuage : DescribeKey ValueStore cloudfront : Politique GetCache front de nuage : GetCache PolicyConfig front de nuage : GetCloud FrontOrigin AccessIdentity cloudfront : Config GetCloud FrontOrigin AccessIdentity front de nuage : GetContinuous DeploymentPolicy cloudfront : Config GetContinuous DeploymentPolicy cloudfront : Config GetDistribution front de nuage : GetField LevelEncryption

Préfixe de service	Actions
	cloudfront : Config GetField LevelEncryption cloudfront : Profil GetField LevelEncryption front de nuage : GetField LevelEncryption ProfileConfig front de nuage : GetFunction front de nuage : GetInvalidation cloudfront : Groupe GetKey front de nuage : GetKey GroupConfig cloudfront : Abonnement GetMonitoring front de nuage : GetOrigin AccessControl cloudfront : Config GetOrigin AccessControl front de nuage : GetOrigin RequestPolicy cloudfront : Config GetOrigin RequestPolicy cloudfront : clé GetPublic front de nuage : GetPublic KeyConfig front de nuage : GetRealtime LogConfig front de nuage : GetResponse HeadersPolicy cloudfront : Config GetResponse HeadersPolicy cloudfront : Distribution GetStreaming front de nuage : GetStreaming DistributionConfig cloudfront : Politiques ListCache front de nuage : ListCloud FrontOrigin AccessIdentities

Préfixe de service	Actions
	cloudfront : Alias ListConflicting front de nuage : ListContinuous DeploymentPolicies front de nuage : ListDistributions front de nuage : ListDistributions ByCache PolicyId cloudfront : Groupe ListDistributions ByKey cloudfront : ID ListDistributions ByOrigin RequestPolicy front de nuage : ListDistributions ByRealtime LogConfig cloudfront : ID ListDistributions ByResponse HeadersPolicy cloudfront : Calid ListDistributions ByWeb cloudfront : Configurations ListField LevelEncryption cloudfront : Profils ListField LevelEncryption front de nuage : ListFunctions front de nuage : ListInvalidations cloudfront : Groupes ListKey front de nuage : ListKey ValueStores front de nuage : ListOrigin AccessControls front de nuage : ListOrigin RequestPolicies cloudfront : Clés ListPublic front de nuage : ListRealtime LogConfigs front de nuage : ListResponse HeadersPolicies cloudfront : Distributions ListStreaming

Préfixe de service	Actions
	front de nuage : PublishFunction front de nuage : TestFunction cloudfront : Politique UpdateCache front de nuage : UpdateCloud FrontOrigin AccessIdentity front de nuage : UpdateContinuous DeploymentPolicy front de nuage : UpdateDistribution cloudfront : Config UpdateField LevelEncryption cloudfront : Profil UpdateField LevelEncryption front de nuage : UpdateFunction cloudfront : Groupe UpdateKey front de nuage : UpdateKey ValueStore front de nuage : UpdateOrigin AccessControl front de nuage : UpdateOrigin RequestPolicy cloudfront : clé UpdatePublic front de nuage : UpdateRealtime LogConfig front de nuage : UpdateResponse HeadersPolicy

Préfixe de service	Actions
cloudhsm	cloudhsm : CreateHapg
	cloudhsm : CreateHsm
	cloudhsm : Client CreateLuna
	cloudhsm : DeleteBackup
	cloudhsm : DeleteHapg
	cloudhsm : DeleteHsm
	cloudhsm : Client DeleteLuna
	cloudhsm : DescribeBackups
	cloudhsm : DescribeClusters
	cloudhsm : DescribeHapg
	cloudhsm : DescribeHsm
	cloudhsm : Client DescribeLuna
	cloudhsm : GetConfig
	cloudhsm : InitializeCluster
	cloudhsm : Zones ListAvailable
	cloudhsm : ListHapgs
	cloudhsm : ListHsms
	cloudhsm : Clientèle ListLuna
	cloudhsm : Attributs ModifyBackup
	cloudhsm : ModifyCluster
	cloudhsm : ModifyHapg

Préfixe de service	Actions
	cloudhsm : ModifyHsm cloudhsm : Client ModifyLuna cloudhsm : RestoreBackup

Préfixe de service	Actions
cloudsearch	recherche dans le cloud : BuildSuggesters recherche dans le cloud : CreateDomain cloudsearch : Schéma DefineAnalysis recherche dans le cloud : DefineExpression cloudsearch : Champ DefineIndex recherche dans le cloud : DefineSuggester cloudsearch : Schéma DeleteAnalysis recherche dans le cloud : DeleteDomain recherche dans le cloud : DeleteExpression cloudsearch : Champ DeleteIndex recherche dans le cloud : DeleteSuggester cloudsearch : Schémas DescribeAnalysis recherche dans le cloud : options DescribeAvailability recherche dans le cloud : DescribeDomain EndpointOptions recherche dans le cloud : DescribeDomains recherche dans le cloud : DescribeExpressions cloudsearch : Champs DescribeIndex cloudsearch : Paramètres DescribeScaling recherche dans le cloud : DescribeService AccessPolicies recherche dans le cloud : DescribeSuggesters recherche dans le cloud : IndexDocuments

Préfixe de service	Actions
	cloudsearch : Noms ListDomain recherche dans le cloud : options UpdateAvailability recherche dans le cloud : UpdateDomain EndpointOptions cloudsearch : Paramètres UpdateScaling recherche dans le cloud : UpdateService AccessPolicies

Préfixe de service	Actions
cloudtrail	traînée nuageuse : CancelQuery
	traînée nuageuse : CreateChannel
	traînée nuageuse : CreateEvent DataStore
	traînée nuageuse : CreateTrail
	traînée nuageuse : DeleteChannel
	traînée nuageuse : DeleteEvent DataStore
	cloudtrail : Politique DeleteResource
	traînée nuageuse : DeleteTrail
	traînée nuageuse : DeregisterOrganization DelegatedAdmin
	traînée nuageuse : DescribeQuery
	traînée nuageuse : DescribeTrails
	traînée nuageuse : DisableFederation
	traînée nuageuse : GetChannel
	traînée nuageuse : GetEvent DataStore
	cloudtrail : Données GetEvent DataStore
	cloudtrail : Sélecteurs GetEvent
	traînée nuageuse : GetImport
	cloudtrail : Sélecteurs GetInsight
	cloudtrail : Résultats GetQuery
	cloudtrail : Politique GetResource
	traînée nuageuse : GetTrail

Préfixe de service	Actions
	cloudtrail : État GetTrail
	traînée nuageuse : ListChannels
	traînée nuageuse : ListEvent DataStores
	cloudtrail : Défaillances ListImport
	traînée nuageuse : ListImports
	cloudtrail : Clés ListPublic
	traînée nuageuse : ListQueries
	traînée nuageuse : ListTrails
	traînée nuageuse : LookupEvents
	cloudtrail : Sélecteurs PutEvent
	cloudtrail : Sélecteurs PutInsight
	cloudtrail : Politique PutResource
	traînée nuageuse : RegisterOrganization DelegatedAdmin
	traînée nuageuse : RestoreEvent DataStore
	cloudtrail : Ingestion StartEvent DataStore
	traînée nuageuse : StartImport
	traînée nuageuse : StartLogging
	traînée nuageuse : StartQuery
	cloudtrail : Ingestion StopEvent DataStore
	traînée nuageuse : StopImport
	traînée nuageuse : StopLogging

Préfixe de service	Actions
	traînée nuageuse : UpdateChannel traînée nuageuse : UpdateEvent DataStore traînée nuageuse : UpdateTrail

Préfixe de service	Actions
cloudwatch	surveillance des nuages : DeleteAlarms cloudwatch : Détecteur DeleteAnomaly surveillance des nuages : DeleteDashboards cloudwatch : Règles DeleteInsight cloudwatch : Stream DeleteMetric cloudwatch : Historique DescribeAlarm surveillance des nuages : DescribeAlarms surveillance des nuages : DescribeAlarms ForMetric cloudwatch : Détecteurs DescribeAnomaly cloudwatch : Règles DescribeInsight cloudwatch : Actions DisableAlarm cloudwatch : Règles DisableInsight cloudwatch : Actions EnableAlarm cloudwatch : Règles EnableInsight surveillance des nuages : GetDashboard surveillance des nuages : GetInsight RuleReport cloudwatch : Stream GetMetric surveillance des nuages : ListDashboards surveillance des nuages : ListManaged InsightRules cloudwatch : Streams ListMetric cloudwatch : Détecteur PutAnomaly

Préfixe de service	Actions
	cloudwatch : Alarme PutComposite surveillance des nuages : PutDashboard cloudwatch : PutInsight Règle surveillance des nuages : PutManaged InsightRules cloudwatch : Alarme PutMetric cloudwatch : Stream PutMetric cloudwatch : État SetAlarm cloudwatch : Streams StartMetric cloudwatch : Streams StopMetric

Préfixe de service	Actions
codeartifact	codeartefact : Connexion AssociateExternal
	artefact de code : Versions CopyPackage
	artefact de code : CreateDomain
	artefact de code : CreateRepository
	artefact de code : DeleteDomain
	artefact de code : DeleteDomain PermissionsPolicy
	artefact de code : DeletePackage
	artefact de code : Versions DeletePackage
	artefact de code : DeleteRepository
	artefact de code : DeleteRepository PermissionsPolicy
	artefact de code : DescribeDomain
	artefact de code : DescribePackage
	artefact de code : Version DescribePackage
	artefact de code : DescribeRepository
	codeartefact : Connexion DisassociateExternal
	artefact de code : Versions DisposePackage
	artefact de code : GetAssociated PackageGroup
	codeartefact : Token GetAuthorization
	artefact de code : GetDomain PermissionsPolicy
	artefact de code : GetPackage VersionAsset
	artefact de code : GetPackage VersionReadme

Préfixe de service	Actions
	codeartefact : Endpoint GetRepository
	artefact de code : GetRepository PermissionsPolicy
	artefact de code : ListDomains
	codeartefact : Groupes ListPackage
	artefact de code : ListPackages
	artefact de code : ListPackage VersionAssets
	artefact de code : ListPackage VersionDependencies
	artefact de code : Versions ListPackage
	artefact de code : ListRepositories
	artefact de code : ListRepositories InDomain
	artefact de code : Version PublishPackage
	artefact de code : PutDomain PermissionsPolicy
	codeartefact : Métadonnées PutPackage
	artefact de code : PutPackage OriginConfiguration
	artefact de code : PutRepository PermissionsPolicy
	codeartifact : Référentiel ReadFrom
	artefact de code : UpdatePackage VersionsStatus
	artefact de code : UpdateRepository

Préfixe de service	Actions
codedeploy	déploiement de code : BatchGet ApplicationRevisions codedeploy : Applications BatchGet déploiement de code : BatchGet DeploymentGroups déploiement de code : BatchGet DeploymentInstances codedeploy : Déploiements BatchGet déploiement de code : BatchGet DeploymentTargets codedeploy : Instances BatchGet OnPremises déploiement de code : ContinueDeployment déploiement de code : CreateApplication déploiement de code : CreateDeployment codedeploy : Config CreateDeployment codedeploy : Groupe CreateDeployment déploiement de code : DeleteApplication codedeploy : Config DeleteDeployment codedeploy : Groupe DeleteDeployment codedeploy : DeleteGit HubAccount jeton codedeploy : ID DeleteResources ByExternal déploiement de code : DeregisterOn PremisesInstance déploiement de code : GetApplication codedeploy : révision GetApplication déploiement de code : GetDeployment

Préfixe de service	Actions
	codedeploy : Config GetDeployment codedeploy : Groupe GetDeployment codedeploy : Instance GetDeployment codedeploy : cible GetDeployment déploiement de code : GetOn PremisesInstance codedeploy : révisions ListApplication déploiement de code : ListApplications codedeploy : Configurations ListDeployment codedeploy : Groupes ListDeployment codedeploy : Instances ListDeployment déploiement de code : ListDeployments codedeploy : cibles ListDeployment déploiement de code : ListGit HubAccount TokenNames déploiement de code : ListOn PremisesInstances déploiement de code : PutLifecycle EventHook ExecutionStatus codedeploy : révision RegisterApplication déploiement de code : RegisterOn PremisesInstance déploiement de code : SkipWait TimeFor InstanceTermination déploiement de code : StopDeployment déploiement de code : UpdateApplication codedeploy : Groupe UpdateDeployment

Préfixe de service	Actions
codeguru-profiler	codeguru-profiler : Chaînes AddNotification
	codeguru-profiler : Données BatchGet FrameMetric
	profileur de code : ConfigureAgent
	codeguru-profiler : Groupe CreateProfiling
	codeguru-profiler : Groupe DeleteProfiling
	codeguru-profiler : Groupe DescribeProfiling
	codeguru-profiler : Résumé GetFindings ReportAccount
	codeguru-profiler : Configuration GetNotification
	profileur de code : GetPolicy
	profileur de code : GetProfile
	profileur de code : GetRecommendations
	codeguru-profiler : Rapports ListFindings
	codeguru-profiler : Times ListProfile
	codeguru-profiler : Groupes ListProfiling
	profileur de code : PutPermission
	codeguru-profiler : Chaîne RemoveNotification
	profileur de code : RemovePermission
	profileur de code : SubmitFeedback
	codeguru-profiler : Groupe UpdateProfiling

Préfixe de service	Actions
codeguru-reviewer	réviseur de code : AssociateRepository codeguru-reviewer : Critique CreateCode codeguru-reviewer : Critique DescribeCode codeguru-reviewer : Commentaires DescribeRecommendation codeguru-reviewer : Association DescribeRepository réviseur de code : DisassociateRepository codeguru-reviewer : Critiques ListCode codeguru-reviewer : Commentaires ListRecommendation réviseur de code : ListRecommendations codeguru-reviewer : Associations ListRepository codeguru-reviewer : Commentaires PutRecommendation

Préfixe de service	Actions
codepipeline	pipeline de code : AcknowledgeJob
	pipeline de code : AcknowledgeThird PartyJob
	pipeline de code : CreateCustom ActionType
	pipeline de code : CreatePipeline
	pipeline de code : DeleteCustom ActionType
	pipeline de code : DeletePipeline
	pipeline de code : DeleteWebhook
	codepipeline : Fête DeregisterWebhook WithThird
	codepipeline : Type GetAction
	codepipeline : Détails GetJob
	pipeline de code : GetPipeline
	codepipeline : Exécution GetPipeline
	codepipeline : État GetPipeline
	codepipeline : Détails GetThird PartyJob
	codepipeline : Exécutions ListAction
	codepipeline : types ListAction
	codepipeline : Exécutions ListPipeline
	pipeline de code : ListPipelines
	pipeline de code : ListWebhooks
	codepipeline : PollFor Offres d'emploi
	codepipeline : PollFor ThirdParty Offres d'emploi

Préfixe de service	Actions
	codepipeline : révision PutAction
	codepipeline : Résultat PutApproval
	pipeline de code : PutJob FailureResult
	pipeline de code : PutJob SuccessResult
	pipeline de code : PutThird PartyJob FailureResult
	pipeline de code : PutThird PartyJob SuccessResult
	pipeline de code : PutWebhook
	codepipeline : Fête RegisterWebhook WithThird
	pipeline de code : RollbackStage
	codepipeline : Exécution StartPipeline
	codepipeline : Exécution StopPipeline
	codepipeline : Type UpdateAction
	pipeline de code : UpdatePipeline

Préfixe de service	Actions
codestar	codestar : Membre AssociateTeam codestar : CreateProject codestar : Profil CreateUser codestar : DeleteProject codestar : Profil DeleteUser codestar : DescribeProject codestar : Profil DescribeUser codestar : Membre DisassociateTeam codestar : ListProjects codestar : ListResources codestar : Membres ListTeam codestar : Profils ListUser codestar : UpdateProject codestar : Membre UpdateTeam codestar : Profil UpdateUser

Préfixe de service	Actions
codestar-notifications	codestar-notifications : Règle CreateNotification
	codestar-notifications : Règle DeleteNotification
	notifications codestar : DeleteTarget
	codestar-notifications : Règle DescribeNotification
	codestar-notifications : types ListEvent
	codestar-notifications : Règles ListNotification
	notifications codestar : ListTargets
	codestar-notifications:Subscribe
	codestar-notifications:Unsubscribe
	codestar-notifications : Règle UpdateNotification

Préfixe de service	Actions
cognito-identity	cognito-identity : Pool CreateIdentity identité cognitive : DeleteIdentities cognito-identity : Pool DeleteIdentity identité cognitive : DescribeIdentity cognito-identity : Pool DescribeIdentity identité cognitive : GetIdentity PoolRoles identité cognitive : ListIdentities cognito-identity : Pools ListIdentity cognito-identity : Identité LookupDeveloper cognito-identity : Identités MergeDeveloper identité cognitive : SetIdentity PoolRoles cognito-identity : Identité UnlinkDeveloper cognito-identity : Pool UpdateIdentity

Préfixe de service	Actions
cognito-idp	cognito-idp : Attributs AddCustom
	cognito-idp : Groupe AdminAdd UserTo
	identifiant : AdminConfirm SignUp
	cognito-idp : utilisateur AdminCreate
	cognito-idp : utilisateur AdminDelete
	identifiant : AdminDelete UserAttributes
	cognito-idp : utilisateur AdminDisable ProviderFor
	cognito-idp : utilisateur AdminDisable
	cognito-idp : utilisateur AdminEnable
	cognito-idp : Appareil AdminForget
	cognito-idp : Appareil AdminGet
	cognito-idp : utilisateur AdminGet
	cognito-idp : Auth AdminInitiate
	cognito-idp : utilisateur AdminLink ProviderFor
	cognito-idp : Appareils AdminList
	cognito-idp : utilisateur AdminList GroupsFor
	cognito-idp : Événements AdminList UserAuth
	cognito-idp : Groupe AdminRemove UserFrom
	identifiant : AdminReset UserPassword
	cognito-idp : Défi AdminRespond ToAuth
	cognito-idp : UserMfapReference AdminSet

Préfixe de service	Actions
	identifiant : AdminSet UserPassword
	identifiant : AdminSet UserSettings
	cognito-idp : Commentaires AdminUpdate AuthEvent
	identifiant : AdminUpdate DeviceStatus
	identifiant : AdminUpdate UserAttributes
	cognito-idp : Indisponible AdminUser GlobalSign
	cognito-idp : jeton AssociateSoftware
	identifiant : ChangePassword
	identifiant : ConfirmDevice
	cognito-idp : Mot de passe ConfirmForgot
	cognito-idp : actif ConfirmSign
	identifiant : CreateGroup
	cognito-idp : Fournisseur CreateIdentity
	cognito-idp : Serveur CreateResource
	identifiant : CreateUser ImportJob
	cognito-idp : Piscine CreateUser
	identifiant : CreateUser PoolClient
	identifiant : CreateUser PoolDomain
	identifiant : DeleteGroup
	cognito-idp : Fournisseur DeleteIdentity
	cognito-idp : Serveur DeleteResource

Préfixe de service	Actions
	identifiant : DeleteUser
	cognito-idp : Attributs DeleteUser
	cognito-idp : Piscine DeleteUser
	identifiant : DeleteUser PoolClient
	identifiant : DeleteUser PoolDomain
	cognito-idp : Fournisseur DescribeIdentity
	cognito-idp : Serveur DescribeResource
	cognito-idp : Configuration DescribeRisk
	identifiant : DescribeUser ImportJob
	cognito-idp : Piscine DescribeUser
	identifiant : DescribeUser PoolClient
	identifiant : DescribeUser PoolDomain
	identifiant : ForgetDevice
	identifiant : ForgotPassword
	cognito-idp:GetCSVHeader
	identifiant : GetDevice
	identifiant : GetGroup
	cognito-idp : Identifiant GetIdentity ProviderBy
	identifiant : GetLog DeliveryConfiguration
	cognito-idp : Certificat GetSigning
	cognito-idp:GetUICustomization

Préfixe de service	Actions
	identifiant : GetUser
	cognito-idp : Code GetUser AttributeVerification
	cognito-idp : Config GetUser PoolMfa
	cognito-idp : Indisponible GlobalSign
	identifiant : InitiateAuth
	identifiant : ListDevices
	identifiant : ListGroups
	cognito-idp : Fournisseurs ListIdentity
	cognito-idp : Serveurs ListResource
	identifiant : ListUser ImportJobs
	identifiant : ListUser PoolClients
	cognito-idp : Piscines ListUser
	identifiant : ListUsers
	identifiant : ListUsers InGroup
	cognito-idp : Code ResendConfirmation
	identifiant : RespondTo AuthChallenge
	identifiant : RevokeToken
	identifiant : SetLog DeliveryConfiguration
	cognito-idp : Configuration SetRisk
	cognito-idp:SetUICustomization
	cognito-idp : MFAPreference SetUser

Préfixe de service	Actions
	cognito-idp : Config SetUser PoolMfa
	cognito-idp : Paramètres SetUser
	identifiant : SignUp
	identifiant : StartUser ImportJob
	identifiant : StopUser ImportJob
	identifiant : UpdateAuth EventFeedback
	cognito-idp : Statut UpdateDevice
	identifiant : UpdateGroup
	cognito-idp : Fournisseur UpdateIdentity
	cognito-idp : Serveur UpdateResource
	cognito-idp : Attributs UpdateUser
	cognito-idp : Piscine UpdateUser
	identifiant : UpdateUser PoolClient
	identifiant : UpdateUser PoolDomain
	cognito-idp : jeton VerifySoftware
	cognito-idp : Attribut VerifyUser

Préfixe de service	Actions
cognito-sync	synchronisation cognitive : BulkPublish synchronisation cognitive : DeleteDataset synchronisation cognitive : DescribeDataset synchronisation cognitive : DescribeIdentity PoolUsage cognito-sync : Utilisation DescribeIdentity synchronisation cognitive : GetBulk PublishDetails cognito-sync : Événements GetCognito synchronisation cognitive : GetIdentity PoolConfiguration synchronisation cognitive : ListDatasets synchronisation cognitive : ListIdentity PoolUsage synchronisation cognitive : ListRecords synchronisation cognitive : RegisterDevice cognito-sync : Événements SetCognito synchronisation cognitive : SetIdentity PoolConfiguration cognito-sync : ensemble de données SubscribeTo cognito-sync : ensemble de données UnsubscribeFrom synchronisation cognitive : UpdateRecords

Préfixe de service	Actions
comprehendmedical	Comprehenmedical : DetectionV2Job DescribeEntities
	Comprehend Medical : Décrivez ICD 10 cm InferenceJob
	Comprehended Medical : Décrivez PHI DetectionJob
	comprehendmedical : Job DescribeRx NormInference
	Comprehend Medical : décrit NomeDCT InferenceJob
	medical complet : V2 DetectEntities
	comprehendmedical:DetectPHI
	comprehendmedical:InferICD10CM
	Comprehendmedical : Norm InferRx
	comprehendmedical:InferSNOMEDCT
	Comprehendmedical : Detection V2jobs ListEntities
	Comprehend Medical : Listic D 10 cm InferenceJobs
	Comprehend Medical : ListPhi DetectionJobs
	comprehendmedical : Offres d'emploi ListRx NormInference
	Comprehend Medical : ne répertorie aucun médicament pour tomodensitométrie InferenceJobs
	Comprehenmedical : DetectionV2Job StartEntities
	Comprehend Medical : Startic D 10 cm InferenceJob
	Comprehend Medical : StartPhi DetectionJob
	comprehendmedical : Job StartRx NormInference
	Comprehend Medical : Démarre la tomodensitométrie InferenceJob

Préfixe de service	Actions
	Comprehenmedical : DetectionV2Job StopEntities Comprehend Medical : STOPIC D 10 cm InferenceJob Comprehend Medical : Stopphi DetectionJob comprehendmedical : Job StopRx NormInference Comprehend Medical : arrête le traitement par tomodensitométrie InferenceJob

Préfixe de service	Actions
compute-optimizer	compute-optimizer : Préférences DeleteRecommendation optimiseur informatique : DescribeRecommendation ExportJobs compute-optimizer : Recommandations ExportAuto ScalingGroup Optimiseur informatique : ExportEBS VolumeRecommendations Optimiseur informatique : Export EC2 InstanceRecommendations Optimiseur informatique : ExportECS ServiceRecommendations optimiseur informatique : ExportLambda FunctionRecommendations compute-optimizer : Recommandations ExportLicense Optimiseur informatique : Getec2 Metrics RecommendationProj ected Optimiseur informatique : GetECS ServiceRecommendation ProjectedMetrics optimiseur informatique : GetEffective RecommendationPreferences compute-optimizer : État GetEnrollment compute-optimizer : Organisation GetEnrollment StatusesFor compute-optimizer : Préférences GetRecommendation compute-optimizer : Résumés GetRecommendation compute-optimizer : Préférences PutRecommendation compute-optimizer : État UpdateEnrollment

Préfixe de service	Actions
config	configuration : BatchGet ResourceConfig
	config : DeleteAggregation Autorisation
	config : DeleteConfig Règle
	config : DeleteConfiguration Aggregator
	config : DeleteConfiguration Enregistreur
	configuration : DeleteConformance Pack
	config : DeleteDelivery Canal
	config : DeleteEvaluation Résultats
	configuration : DeleteOrganization ConfigRule
	configuration : DeleteOrganization ConformancePack
	configuration : DeletePending AggregationRequest
	config : DeleteRemediation Configuration
	configuration : DeleteRemediation Exceptions
	config : DeleteResource Config
	config : DeleteRetention Configuration
	config : DeleteStored Requête
	configuration : DeliverConfig Snapshot
	configuration : DescribeAggregate ComplianceBy ConfigRules
	configuration : DescribeAggregate ComplianceBy Conforman cePacks
	config : DescribeAggregation Autorisations

Préfixe de service	Actions
	config : DescribeCompliance ByConfig Règle
	configuration : DescribeCompliance ByResource
	config : DescribeConfig RuleEvaluation État
	config : DescribeConfig Règles
	config : DescribeConfiguration Agrégateurs
	config : DescribeConfiguration AggregatorSources État
	config : DescribeConfiguration Enregistreurs
	configuration : DescribeConfiguration RecorderStatus
	configuration : DescribeConformance PackCompliance
	configuration : DescribeConformance Packs
	configuration : DescribeConformance PackStatus
	config : DescribeDelivery Canaux
	configuration : DescribeDelivery ChannelStatus
	configuration : DescribeOrganization ConfigRules
	config : DescribeOrganization ConfigRule Statuts
	configuration : DescribeOrganization ConformancePacks
	config : DescribeOrganization ConformancePack Statuts
	configuration : DescribePending AggregationRequests
	config : DescribeRemediation Configurations
	configuration : DescribeRemediation Exceptions
	configuration : DescribeRemediation ExecutionStatus

Préfixe de service	Actions
	config : DescribeRetention Configurations
	configuration : GetCompliance DetailsBy ConfigRule
	config : GetCompliance DetailsBy Ressource
	configuration : GetCompliance SummaryBy ConfigRule
	configuration : GetCompliance SummaryBy ResourceType
	config : GetConformance PackCompliance Détails
	config : GetConformance PackCompliance Résumé
	configuration : GetCustom RulePolicy
	configuration : GetDiscovered ResourceCounts
	configuration : GetOrganization ConfigRule DetailedStatus
	configuration : GetOrganization ConformancePack DetailedStatus
	config : GetOrganization CustomRule Politique
	configuration : GetResource ConfigHistory
	configuration : GetResource EvaluationSummary
	config : GetStored Requête
	config : ListConformance PackCompliance Scores
	config : ListDiscovered Ressources
	config : ListResource Évaluations
	config : ListStored Requêtes
	config : PutConfig Règle
	config : PutConfiguration Aggregator

Préfixe de service	Actions
	config : PutConfiguration Enregistreur
	configuration : PutConformance Pack
	config : PutDelivery Canal
	configuration : PutEvaluations
	config : PutExternal Évaluation
	configuration : PutOrganization ConfigRule
	configuration : PutOrganization ConformancePack
	config : PutRemediation Configurations
	configuration : PutRemediation Exceptions
	config : PutResource Config
	config : PutRetention Configuration
	config : PutStored Requête
	config : SelectResource Config
	configuration : StartConfig RulesEvaluation
	config : StartConfiguration Enregistreur
	config : StartRemediation Exécution
	config : StartResource Évaluation
	config : StopConfiguration Enregistreur

Préfixe de service	Actions
connect	connect : ActivateEvaluation Form connecter : AssociateApproved Origin connecter : AssociateBot connect : AssociateDefault Vocabulaire connecter : AssociateFlow connecter : AssociateInstance StorageConfig connecter : AssociateLambda Fonction connecter : AssociateLex Bot connecter : AssociatePhone NumberContact Flow connecter : AssociateQueue QuickConnects connecter : AssociateRouting ProfileQueues connecter : AssociateSecurity clé connect : AssociateUser Proficiencies connecter : BatchGet FlowAssociation connect : BatchPut Contact connecter : ClaimPhone Numéro connect : CreateAgent Status connecter : CreateContact Flow connecter : CreateContact FlowModule connect : CreateEvaluation Form connecter : CreateHours OfOperation

Préfixe de service	Actions
	connecter : CreateInstance connecter : CreateIntegration Association connecter : CreateParticipant connecter : CreatePersistent ContactAssociation connect : CreatePredefined Attribut connecter : CreatePrompt connecter : CreateQueue connecter : CreateQuick Connect connect : CreateRouting Profil connecter : CreateRule connect : CreateSecurity Profil connect : CreateTask Modèle connecter : CreateTraffic DistributionGroup Connect : CreateUse Étui connecter : CreateUser connecter : CreateUser HierarchyGroup connecter : CreateView connecter : CreateView Version connecter : CreateVocabulary connect : DeactivateEvaluation Form connect : DeleteContact Évaluation

Préfixe de service	Actions
	connecter : DeleteContact Flow connecter : DeleteContact FlowModule connect : DeleteEvaluation Form connecter : DeleteHours OfOperation connecter : DeleteInstance connecter : DeleteIntegration Association connect : DeletePredefined Attribut connecter : DeletePrompt connecter : DeleteQueue connecter : DeleteQuick Connect connect : DeleteRouting Profil connecter : DeleteRule connect : DeleteSecurity Profil connect : DeleteTask Modèle connecter : DeleteTraffic DistributionGroup Connect : DeleteUse Étui connecter : DeleteUser connecter : DeleteUser HierarchyGroup connecter : DeleteView connecter : DeleteVocabulary connect : DescribeAgent Status

Préfixe de service	Actions
	connect : DescribeContact Évaluation connecter : DescribeContact Flow connecter : DescribeContact FlowModule connect : DescribeEvaluation Form connect : DescribeInstance Attribut connecter : DescribeInstance StorageConfig connecter : DescribePhone Numéro connecter : DescribeRule connecter : DescribeTraffic DistributionGroup connecter : DescribeUser HierarchyGroup connecter : DescribeUser HierarchyStructure connecter : DescribeView connecter : DescribeVocabulary connecter : DisassociateApproved Origin connecter : DisassociateBot connecter : DisassociateFlow connecter : DisassociateInstance StorageConfig connecter : DisassociateLambda Fonction connecter : DisassociateLex Bot connecter : DisassociatePhone NumberContact Flow connecter : DisassociateQueue QuickConnects

Préfixe de service	Actions
	connecter : DisassociateRouting ProfileQueues connecter : DisassociateSecurity clé connect : DisassociateUser Proficiencies connect : DismissUser Contact connect : GetContact Attributs connecter : GetCurrent MetricData connecter : GetCurrent UserData connecter : GetFederation Token connecter : GetFlow Association connecter : GetMetric Data connecter : GetMetric DataV2 connect : GetPrompt Fichier connect : GetTask Modèle connecter : GetTraffic Distribution connecter : ImportPhone Numéro connecter : ListApproved Origins connecter : ListBots connect : ListContact Évaluations connecter : ListContact FlowModules connecter : ListContact Flows connect : ListContact Références

Préfixe de service	Actions
	connect : ListDefault Vocabulaires connect : ListEvaluation Formulaires connecter : ListEvaluation FormVersions connecter : ListFlow Associations connecter : ListHours OfOperations connect : ListInstance Attributs connecter : ListInstance StorageConfigs connecter : ListIntegration Associations connect : ListLambda Fonctions connecter : ListLex Bots connecter : ListPhone Numbers connecter : ListPhone NumberSV2 connect : ListPredefined Attributs connecter : ListPrompts connecter : ListQueue QuickConnects connecter : ListQueues connecter : ListQuick Connecte connecter : ListRealtime ContactAnalysis SegmentSv2 connecter : ListRouting ProfileQueues connect : ListRouting Profilés connecter : ListRules

Préfixe de service	Actions
	connecter : ListSecurity Clés connecter : ListSecurity ProfileApplications connecter : ListSecurity ProfilePermissions connect : ListSecurity Profilés connect : ListTask Modèles connecter : ListTraffic DistributionGroups connect : ListUse Cases connecter : ListUser HierarchyGroups connect : ListUser Proficiencies connecter : ListUsers connecter : ListViews connecter : ListView Versions connecter : MonitorContact connecter : PauseContact connect : PutUser Status connecter : ReleasePhone Numéro connecter : ReplicateInstance connecter : ResumeContact connecter : ResumeContact Enregistrement connecter : SearchAvailable PhoneNumbers connecter : SearchContacts

Préfixe de service	Actions
	connecter : SearchHours OfOperations connect : SearchPredefined Attributs connecter : SearchPrompts connecter : SearchQueues connecter : SearchQuick Connecte connect : SearchRouting Profilés connect : SearchSecurity Profilés connecter : SearchVocabularies connecter : SendChat IntegrationEvent connect : StartChat Contact connect : StartContact Évaluation connecter : StartContact Enregistrement connecter : StartContact Streaming connecter : StartOutbound VoiceContact connect : StartTask Contact connecter : StartWeb RTCcontact connecter : StopContact connecter : StopContact Enregistrement connecter : StopContact Streaming connect : SubmitContact Évaluation connecter : SuspendContact Enregistrement

Préfixe de service	Actions
	connecter : TransferContact connect : UpdateAgent Status connecter : UpdateContact connect : UpdateContact Attributs connect : UpdateContact Évaluation connecter : UpdateContact FlowContent connecter : UpdateContact FlowMetadata connect : UpdateContact FlowModule Contenu connect : UpdateContact FlowModule Metadata connecter : UpdateContact FlowName connecter : UpdateContact RoutingData connect : UpdateContact Calendrier connect : UpdateEvaluation Form connecter : UpdateHours OfOperation connect : UpdateInstance Attribut connecter : UpdateInstance StorageConfig connecter : UpdateParticipant RoleConfig connecter : UpdatePhone Numéro connecter : UpdatePhone NumberMetadata connect : UpdatePredefined Attribut connecter : UpdatePrompt

Préfixe de service	Actions
	connect : UpdateQueue HoursOf Fonctionnement connecter : UpdateQueue MaxContacts connect : UpdateQueue Nom connecter : UpdateQueue OutboundCaller Config connect : UpdateQueue Status connecter : UpdateQuick ConnectConfig connecter : UpdateQuick ConnectName connecter : UpdateRouting ProfileAgent AvailabilityTimer connecter : UpdateRouting ProfileConcurrency connecter : UpdateRouting ProfileDefault OutboundQueue connecter : UpdateRouting ProfileName connecter : UpdateRouting ProfileQueues connecter : UpdateRule connect : UpdateSecurity Profil connect : UpdateTask Modèle connecter : UpdateTraffic Distribution connect : UpdateUser Hiérarchie connect : UpdateUser HierarchyGroup Nom connecter : UpdateUser HierarchyStructure connecter : UpdateUser IdentityInfo connecter : UpdateUser PhoneConfig

Préfixe de service	Actions
	connect : UpdateUser Proficiencies connecter : UpdateUser RoutingProfile connecter : UpdateUser SecurityProfiles connect : UpdateView Contenu connect : UpdateView Metadata
cur	cur : DeleteReport Définition cur : DescribeReport Définitions cur : ModifyReport Définition cur : PutReport Définition

Préfixe de service	Actions
databrew	brassage de données : BatchDelete RecipeVersion base de données : CreateDataset databrew : Job CreateProfile base de données : CreateProject base de données : CreateRecipe databrew : Job CreateRecipe base de données : CreateRuleset base de données : CreateSchedule base de données : DeleteDataset base de données : DeleteJob base de données : DeleteProject databrew : Version DeleteRecipe base de données : DeleteRuleset base de données : DeleteSchedule base de données : DescribeDataset base de données : DescribeJob databrew : Exécuter DescribeJob base de données : DescribeProject base de données : DescribeRecipe base de données : DescribeRuleset base de données : DescribeSchedule

Préfixe de service	Actions
	base de données : ListDatasets databrew : Exécute ListJob base de données : ListJobs base de données : ListProjects base de données : ListRecipes databrew : Versions ListRecipe base de données : ListRulesets base de données : ListSchedules base de données : PublishRecipe brassage de données : SendProject SessionAction databrew : Exécuter StartJob databrew : Session StartProject databrew : Exécuter StopJob base de données : UpdateDataset databrew : Job UpdateProfile base de données : UpdateProject base de données : UpdateRecipe databrew : Job UpdateRecipe base de données : UpdateRuleset base de données : UpdateSchedule

Préfixe de service	Actions
dataexchange	échange de données : CancelJob
	échange de données : Set CreateData
	échange de données : Action CreateEvent
	échange de données : CreateJob
	échange de données : CreateRevision
	échange de données : DeleteAsset
	échange de données : Action DeleteEvent
	échange de données : DeleteRevision
	échange de données : Action GetEvent
	échange de données : GetJob
	échange de données : ListData SetRevisions
	échange de données : ensembles ListData
	échange de données : Actions ListEvent
	échange de données : ListJobs
	échange de données : actifs ListRevision
	échange de données : RevokeRevision
	échange de données : SendData SetNotification
	échange de données : StartJob
	échange de données : UpdateAsset
	échange de données : Set UpdateData
	échange de données : Action UpdateEvent

Préfixe de service	Actions
	échange de données : UpdateRevision
datapipeline	pipeline de données : ActivatePipeline pipeline de données : CreatePipeline pipeline de données : DeactivatePipeline pipeline de données : DeletePipeline pipeline de données : DescribeObjects pipeline de données : DescribePipelines pipeline de données : EvaluateExpression datapipeline : définition GetPipeline pipeline de données : ListPipelines datapipeline : Tâche PollFor datapipeline : définition PutPipeline pipeline de données : QueryObjects datapipeline : Progression ReportTask pipeline de données : ReportTask RunnerHeartbeat pipeline de données : SetStatus datapipeline : Status SetTask datapipeline : définition ValidatePipeline

Préfixe de service	Actions
dax	télécopieur : CreateCluster dax : Factor DecreaseReplication télécopieur : DeleteCluster dax : Groupe DeleteParameter dax : Groupe DeleteSubnet télécopieur : DescribeClusters dax : Paramètres DescribeDefault télécopieur : DescribeEvents dax : Groupes DescribeParameter télécopieur : DescribeParameters dax : Groupes DescribeSubnet dax : Factor IncreaseReplication télécopieur : RebootNode télécopieur : UpdateCluster dax : Groupe UpdateParameter dax : Groupe UpdateSubnet

Préfixe de service	Actions
devicefarm	devicefarm : Piscine CreateDevice devicefarm : Profil CreateInstance devicefarm : Profil CreateNetwork ferme d'appareils : CreateProject ferme d'appareils : CreateRemote AccessSession ferme d'appareils : CreateTest GridProject ferme d'appareils : CreateTest GridUrl ferme d'appareils : CreateUpload devicefarm:CreateVPCEConfiguration devicefarm : Piscine DeleteDevice devicefarm : Profil DeleteInstance devicefarm : Profil DeleteNetwork ferme d'appareils : DeleteProject ferme d'appareils : DeleteRemote AccessSession ferme d'appareils : DeleteRun ferme d'appareils : DeleteTest GridProject ferme d'appareils : DeleteUpload devicefarm:DeleteVPCEConfiguration devicefarm : Paramètres GetAccount ferme d'appareils : GetDevice devicefarm : Instance GetDevice

Préfixe de service	Actions
	devicefarm : Piscine GetDevice ferme d'appareils : GetDevice PoolCompatibility devicefarm : Profil GetInstance ferme d'appareils : GetJob devicefarm : Profil GetNetwork devicefarm : État GetOffering ferme d'appareils : GetProject ferme d'appareils : GetRemote AccessSession ferme d'appareils : GetRun ferme d'appareils : GetSuite ferme d'appareils : GetTest ferme d'appareils : GetTest GridProject ferme d'appareils : GetTest GridSession ferme d'appareils : GetUpload devicefarm:GetVPCEConfiguration ferme d'appareils : ListArtifacts devicefarm : Instances ListDevice devicefarm : Pools ListDevice ferme d'appareils : ListDevices devicefarm : Profils ListInstance ferme d'appareils : ListJobs

Préfixe de service	Actions
	devicefarm : Profils ListNetwork devicefarm : Promotions ListOffering ferme d'appareils : ListOfferings devicefarm : Transactions ListOffering ferme d'appareils : ListProjects ferme d'appareils : ListRemote AccessSessions ferme d'appareils : ListRuns ferme d'appareils : ListSamples ferme d'appareils : ListSuites ferme d'appareils : ListTest GridProjects devicefarm : Actions ListTest GridSession devicefarm : Artefacts ListTest GridSession ferme d'appareils : ListTest GridSessions ferme d'appareils : ListTests devicefarm : Problèmes ListUnique ferme d'appareils : ListUploads devicefarm:ListVPCEConfigurations ferme d'appareils : PurchaseOffering ferme d'appareils : RenewOffering ferme d'appareils : ScheduleRun ferme d'appareils : StopJob

Préfixe de service	Actions
	ferme d'appareils : StopRemote AccessSession ferme d'appareils : StopRun devicefarm : Instance UpdateDevice devicefarm : Piscine UpdateDevice devicefarm : Profil UpdateInstance devicefarm : Profil UpdateNetwork ferme d'appareils : UpdateProject ferme d'appareils : UpdateTest GridProject ferme d'appareils : UpdateUpload devicefarm:UpdateVPCEConfiguration

Préfixe de service	Actions
devops-guru	devops-guru : Chaîne AddNotification
	gourou de DevOps : DeleteInsight
	devops-guru : Health DescribeAccount
	devops-guru : Présentation DescribeAccount
	gourou de DevOps : DescribeAnomaly
	gourou de DevOps : DescribeEvent SourcesConfig
	gourou de DevOps : DescribeFeedback
	gourou de DevOps : DescribeInsight
	devops-guru : Health DescribeOrganization
	devops-guru : Présentation DescribeOrganization
	devops-guru : Health DescribeOrganization ResourceCollection
	gourou de DevOps : DescribeResource CollectionHealth
	devops-guru : Intégration DescribeService
	devops-guru : Estimation GetCost
	devops-guru : Collection GetResource
	gourou de DevOps : ListAnomalies ForInsight
	gourou de DevOps : ListAnomalous LogGroups
	gourou de DevOps : ListEvents
	gourou de DevOps : ListInsights
	devops-guru : Ressources ListMonitored
	devops-guru : Chaînes ListNotification

Préfixe de service	Actions
	devops-guru : Perspectives ListOrganization
	gourou de DevOps : ListRecommendations
	gourou de DevOps : PutFeedback
	devops-guru : Chaîne RemoveNotification
	gourou de DevOps : SearchInsights
	devops-guru : Perspectives SearchOrganization
	devops-guru : Estimation StartCost
	gourou de DevOps : UpdateEvent SourcesConfig
	devops-guru : Collection UpdateResource
	devops-guru : Intégration UpdateService

Préfixe de service	Actions
directconnect	connexion directe : AcceptDirect ConnectGateway Associati onProposal connexion directe : AllocateConnection OnInterconnect directconnect : AllocateHosted Connexion connexion directe : AllocatePrivate VirtualInterface connexion directe : AllocatePublic VirtualInterface connexion directe : AllocateTransit VirtualInterface connexion directe : AssociateConnection WithLag directconnect : AssociateHosted Connexion connexion directe : AssociateMac SecKey connexion directe : Interface AssociateVirtual connexion directe : ConfirmConnection directconnect : accord ConfirmCustomer connexion directe : ConfirmPrivate VirtualInterface connexion directe : ConfirmPublic VirtualInterface connexion directe : ConfirmTransit VirtualInterface directconnect:CreateBGPPeer connexion directe : CreateConnection connexion directe : CreateDirect ConnectGateway directconnect : Association CreateDirect ConnectGateway connexion directe : CreateDirect ConnectGateway Associati onProposal

Préfixe de service	Actions
	connexion directe : CreateInterconnect connexion directe : CreateLag connexion directe : CreatePrivate VirtualInterface connexion directe : CreatePublic VirtualInterface connexion directe : CreateTransit VirtualInterface directconnect:DeleteBGPPeer connexion directe : DeleteConnection connexion directe : DeleteDirect ConnectGateway directconnect : Association DeleteDirect ConnectGateway connexion directe : DeleteDirect ConnectGateway Associati onProposal connexion directe : DeleteInterconnect connexion directe : DeleteLag connexion directe : Interface DeleteVirtual connexion directe : DescribeConnection Loa connexion directe : DescribeConnections connexion directe : DescribeConnections OnInterconnect directconnect : métadonnées DescribeCustomer connexion directe : DescribeDirect ConnectGateway Associati onProposals directconnect : Associations DescribeDirect ConnectGateway directconnect : Pièces jointes DescribeDirect ConnectGateway

Préfixe de service	Actions
	connexion directe : DescribeDirect ConnectGateways directconnect : Connexions DescribeHosted connexion directe : DescribeInterconnect Loa connexion directe : DescribeInterconnects connexion directe : DescribeLags connexion directe : DescribeLoa connexion directe : DescribeLocations connexion directe : Configuration DescribeRouter directconnect : Passerelles DescribeVirtual connexion directe : Interfaces DescribeVirtual connexion directe : DisassociateConnection FromLag connexion directe : DisassociateMac SecKey directconnect : Historique ListVirtual InterfaceTest connexion directe : StartBgp FailoverTest connexion directe : StopBgp FailoverTest connexion directe : UpdateConnection connexion directe : UpdateDirect ConnectGateway directconnect : Association UpdateDirect ConnectGateway connexion directe : UpdateLag connexion directe : UpdateVirtual InterfaceAttributes

Préfixe de service	Actions
dlm	dlm : Politique CreateLifecycle
	dlm : Politique DeleteLifecycle
	dlm : Politiques GetLifecycle
	dlm : Politique GetLifecycle
	dlm : Politique UpdateLifecycle

Préfixe de service	Actions
dms	dms : ApplyPending MaintenanceAction dms : Recommandations BatchStart dms : Exécuter CancelReplication TaskAssessment dms : Prestataire CreateData dms : CreateEndpoint dms : Abonnement CreateEvent dms : Profil CreateInstance dms : Projet CreateMigration dms : Config CreateReplication dms : Instance CreateReplication dms : CreateReplication SubnetGroup dms : Tâche CreateReplication dms : DeleteCertificate dms : DeleteConnection dms : Prestataire DeleteData dms : DeleteEndpoint dms : Abonnement DeleteEvent dms : DeleteFleet AdvisorCollector dms : DeleteFleet AdvisorDatabases dms : Profil DeleteInstance dms : Projet DeleteMigration

Préfixe de service	Actions
	dms : Config DeleteReplication dms : Instance DeleteReplication dms : DeleteReplication SubnetGroup dms : Tâche DeleteReplication dms : Exécuter DeleteReplication TaskAssessment dms : Attributs DescribeAccount dms : DescribeApplicable IndividualAssessments dms : DescribeCertificates dms : DescribeConnections dms : DescribeEndpoints dms : Réglages DescribeEndpoint dms : Types DescribeEndpoint dms : Versions DescribeEngine dms : Catégories DescribeEvent dms : DescribeEvents dms : Abonnements DescribeEvent dms : DescribeFleet AdvisorCollectors dms : DescribeFleet AdvisorDatabases dms : Analyse DescribeFleet AdvisorLsa dms : DescribeFleet AdvisorSchema ObjectSummary dms : DescribeFleet AdvisorSchemas

Préfixe de service	Actions
	dms : DescribeMetadata ModelImports dms : DescribeOrderable ReplicationInstances dms : DescribePending MaintenanceActions dms : Limites DescribeRecommendation dms : DescribeRecommendations dms : DescribeRefresh SchemasStatus dms : Configurations DescribeReplication dms : Instances DescribeReplication dms : DescribeReplication InstanceTask Logs dms : DescribeReplications dms : DescribeReplication SubnetGroups dms : DescribeReplication TableStatistics dms : Résultats DescribeReplication TaskAssessment dms : Runs DescribeReplication TaskAssessment dms : Évaluations DescribeReplication TaskIndividual dms : Tâches DescribeReplication dms : DescribeSchemas dms : Statistiques DescribeTable dms : ExportMetadata ModelAssessment dms : modèle GetMetadata dms : ImportCertificate

Préfixe de service	Actions
	dms : ListMetadata ModelAssessment ActionItems dms : ModifyEndpoint dms : Abonnement ModifyEvent dms : Config ModifyReplication dms : Instance ModifyReplication dms : ModifyReplication SubnetGroup dms : Tâche ModifyReplication dms : Tâche MoveReplication dms : Instance RebootReplication dms : RefreshSchemas dms : Tableaux ReloadReplication dms : ReloadTables dms : Analyse RunFleet AdvisorLsa dms : StartMetadata ModelAssessment dms : StartMetadata ModelConversion dms : StartMetadata ModelExport ToTarget dms : StartRecommendations dms : StartReplication dms : Tâche StartReplication dms : StartReplication TaskAssessment dms : Tâche StopReplication

Préfixe de service	Actions
	dms : TestConnection dms : Pont UpdateSubscriptions ToEvent
docdb-elastic	docdb-elastic : Instantané CopyCluster docdb-elastic : DeleteCluster docdb-elastic : Instantané DeleteCluster docdb-elastic : GetCluster docdb-elastic : Instantané GetCluster docdb-elastic : ListClusters docdb-elastic : Instantanés ListCluster docdb-elastic : RestoreCluster FromSnapshot docdb-elastic : StartCluster docdb-elastic : StopCluster docdb-elastic : UpdateCluster

Préfixe de service	Actions
ds	ds : AcceptShared Annuaire Annonces : AddIp Routes annonces : AddRegion annonces : CancelSchema Extension annonces : ConnectDirectory annonces : CreateAlias annonces : CreateComputer ds : CreateConditional Forwarder annonces : CreateDirectory annonces : CreateLog Abonnement Annonces : CreateMicrosoft CAD annonces : CreateSnapshot annonces : CreateTrust ds : DeleteConditional Forwarder annonces : DeleteDirectory annonces : DeleteLog Abonnement annonces : DeleteSnapshot annonces : DeleteTrust annonces : DeregisterCertificate annonces : DeregisterEvent Sujet annonces : DescribeCertificate

Préfixe de service	Actions
	annonces : DescribeClient AuthenticationSettings ds : DescribeConditional Forwarders annonces : DescribeDirectories ds : DescribeDomain Contrôleurs annonces : DescribeEvent Sujets ds:DescribeLDAPSSettings annonces : DescribeRegions annonces : DescribeSettings ds : DescribeShared Répertoires annonces : DescribeSnapshots annonces : DescribeTrusts ds : DescribeUpdate Annuaire ds : DisableClient Authentification ds:DisableLDAPS annonces : DisableRadius annonces : DisableSso ds : EnableClient Authentification ds:EnableLDAPS annonces : EnableRadius annonces : EnableSso annonces : GetDirectory Limites

Préfixe de service	Actions
	annonces : GetSnapshot Limites annonces : ListCertificates Annonces : ListIp Routes annonces : ListLog Abonnements annonces : ListSchema Extensions annonces : RegisterCertificate annonces : RegisterEvent Sujet ds : RejectShared Annuaire Annonces : RemoveIp Routes annonces : RemoveRegion ds : ResetUser Mot de passe annonces : RestoreFrom Snapshot annonces : ShareDirectory annonces : StartSchema Extension annonces : UnshareDirectory ds : UpdateConditional Forwarder ds : UpdateDirectory Configuration ds : UpdateNumber OfDomain Contrôleurs annonces : UpdateRadius annonces : UpdateSettings annonces : UpdateTrust

Préfixe de service	Actions
	annonces : VerifyTrust

Préfixe de service	Actions
dynamodb	dynamodb : CreateBackup
	dynamodb : Tableau CreateGlobal
	dynamodb : CreateTable
	dynamodb : DeleteBackup
	dynamodb : DeleteTable
	dynamodb : DescribeBackup
	dynamodb : Sauvegardes DescribeContinuous
	dynamodb : Perspectives DescribeContributor
	dynamodb : DescribeEndpoints
	dynamodb : DescribeExport
	dynamodb : Tableau DescribeGlobal
	dynamodb : DescribeGlobal TableSettings
	dynamodb : DescribeImport
	dynamodb : DescribeKinesis StreamingDestination
	dynamodb : DescribeLimits
	dynamodb : DescribeStream
	dynamodb : DescribeTable
	dynamodb : Mise à l'échelle DescribeTable ReplicaAuto
	dynamodb : DescribeTime ToLive
	dynamodb : DisableKinesis StreamingDestination
	dynamodb : EnableKinesis StreamingDestination

Préfixe de service	Actions
	dynamodb : ExportTable ToPoint InTime
	dynamodb : Politique GetResource
	dynamodb : ImportTable
	dynamodb : ListBackups
	dynamodb : Perspectives ListContributor
	dynamodb : ListExports
	dynamodb : Tableaux ListGlobal
	dynamodb : ListImports
	dynamodb : ListStreams
	dynamodb : ListTables
	dynamodb : RestoreTable FromBackup
	dynamodb : RestoreTable ToPoint InTime
	dynamodb : Sauvegardes UpdateContinuous
	dynamodb : Perspectives UpdateContributor
	dynamodb : Tableau UpdateGlobal
	dynamodb : UpdateGlobal TableSettings
	dynamodb : UpdateKinesis StreamingDestination
	dynamodb : UpdateTable
	dynamodb : Mise à l'échelle UpdateTable ReplicaAuto
	dynamodb : UpdateTime ToLive

Préfixe de service	Actions
ebs	ebs : CompleteSnapshot ebs : StartSnapshot

Préfixe de service	Actions
ec2	ec2 : Transfert AcceptAddress
	ec2 : AcceptReserved InstancesExchange Citation
	EC2 : AcceptTransit GatewayMulticast DomainAssociations
	ec2 : Pièce jointe AcceptTransit GatewayPeering
	ec2 : Pièce jointe AcceptTransit GatewayVpc
	EC2 : AcceptVpc EndpointConnections
	EC2 : AcceptVpc PeeringConnection
	ec2 : AdvertiseByoip Cidre
	EC2 : AllocateAddress
	EC2 : AllocateHosts
	EC2 : AllocateIpam PoolCidr
	EC2 : ApplySecurity GroupsTo ClientVpn TargetNetwork
	ec2 : 6 Adresses AssignIpv
	EC2 : AssignPrivate IpAddresses
	ec2 : Adresse AssignPrivate NatGateway
	EC2 : AssociateAddress
	ec2 : Réseau AssociateClient VpnTarget
	ec2 : Options AssociateDhcp
	ec2 : Rôle AssociateEnclave CertificateIam
	EC2 : AssociateIam InstanceProfile
	EC2 : AssociateInstance EventWindow

Préfixe de service	Actions
	ec2 : Byosan AssociateIpam
	EC2 : AssociateIpam ResourceDiscovery
	EC2 : AssociateNat GatewayAddress
	ec2 : Tableau AssociateRoute
	EC2 : AssociateSubnet CidrBlock
	ec2 : Domaine AssociateTransit GatewayMulticast
	ec2 : Tableau AssociateTransit GatewayPolicy
	ec2 : Tableau AssociateTransit GatewayRoute
	ec2 : Interface AssociateTrunk
	EC2 : AssociateVpc CidrBlock
	EC2 : AttachClassic LinkVpc
	ec2 : Passerelle AttachInternet
	ec2 : Interface AttachNetwork
	ec2 : Fournisseur AttachVerified AccessTrust
	EC2 : AttachVolume
	ec2 : Passerelle AttachVpn
	EC2 : AuthorizeClient VpnIngress
	EC2 : AuthorizeSecurity GroupEgress
	EC2 : AuthorizeSecurity GroupIngress
	EC2 : BundleInstance
	ec2 : Tâche CancelBundle

Préfixe de service	Actions
	ec2 : Réservation CancelCapacity
	EC2 : CancelCapacity ReservationFleets
	ec2 : Tâche CancelConversion
	ec2 : Tâche CancelExport
	EC2 : CancelImage LaunchPermission
	ec2 : Tâche CancelImport
	EC2 : CancelReserved InstancesListing
	EC2 : CancelSpot FleetRequests
	EC2 : CancelSpot InstanceRequests
	ec2 : Instance ConfirmProduct
	ec2 : Image CopyFpga
	EC2 : CopyImage
	EC2 : CopySnapshot
	ec2 : Réservation CreateCapacity
	EC2 : CreateCapacity ReservationFleet
	ec2 : Passerelle CreateCarrier
	EC2 : CreateClient VpnEndpoint
	EC2 : CreateClient VpnRoute
	ec2 : CreateCoip Cidre
	ec2 : Piscine CreateCoip
	ec2 : Passerelle CreateCustomer

Préfixe de service	Actions
	ec2 : Sous-réseau CreateDefault
	ec2 : VPC CreateDefault
	ec2 : Options CreateDhcp
	ec2 : Passerelle CreateEgress OnlyInternet
	EC2 : CreateFleet
	ec2 : CreateFlow Journaux
	ec2 : Image CreateFpga
	EC2 : CreateImage
	EC2 : CreateInstance ConnectEndpoint
	EC2 : CreateInstance EventWindow
	EC2 : CreateInstance ExportTask
	ec2 : Passerelle CreateInternet
	EC2 : CreateIpam
	ec2 : Piscine CreateIpam
	EC2 : CreateIpam ResourceDiscovery
	ec2 : CreateIpam Champ d'application
	ec2 : paire CreateKey
	ec2 : Modèle CreateLaunch
	EC2 : CreateLaunch TemplateVersion
	EC2 : CreateLocal GatewayRoute
	ec2 : Tableau CreateLocal GatewayRoute

Préfixe de service	Actions
	ec2 : Association CreateLocal GatewayRoute TableVirtual Interface Group
	ec2 : Association CreateLocal GatewayRoute TableVpc
	EC2 : CreateManaged PrefixList
	ec2 : Passerelle CreateNat
	ec2 : Acl CreateNetwork
	EC2 : CreateNetwork AclEntry
	ec2 : CreateNetwork InsightsAccess Champ d'application
	EC2 : CreateNetwork InsightsPath
	ec2 : Interface CreateNetwork
	EC2 : CreateNetwork InterfacePermission
	ec2 : Groupe CreatePlacement
	ec2 : IPv4 Pool CreatePublic
	ec2 : Tâche CreateReplace RootVolume
	EC2 : CreateReserved InstancesListing
	EC2 : CreateRestore ImageTask
	EC2 : CreateRoute
	ec2 : Tableau CreateRoute
	ec2 : Groupe CreateSecurity
	EC2 : CreateSnapshot
	EC2 : CreateSnapshots

Préfixe de service	Actions
	EC2 : CreateSpot DatafeedSubscription
	EC2 : CreateStore ImageTask
	EC2 : CreateSubnet
	EC2 : CreateSubnet CidrReservation
	EC2 : CreateTraffic MirrorFilter
	ec2 : CreateTraffic MirrorFilter Règle
	EC2 : CreateTraffic MirrorSession
	EC2 : CreateTraffic MirrorTarget
	ec2 : Passerelle CreateTransit
	EC2 : CreateTransit GatewayConnect
	ec2 : pair CreateTransit GatewayConnect
	ec2 : Domaine CreateTransit GatewayMulticast
	ec2 : Pièce jointe CreateTransit GatewayPeering
	ec2 : Tableau CreateTransit GatewayPolicy
	EC2 : CreateTransit GatewayPrefix ListReference
	EC2 : CreateTransit GatewayRoute
	ec2 : Tableau CreateTransit GatewayRoute
	EC2 : CreateTransit GatewayRoute TableAnnouncement
	ec2 : Pièce jointe CreateTransit GatewayVpc
	EC2 : CreateVerified AccessEndpoint
	EC2 : CreateVerified AccessGroup

Préfixe de service	Actions
	EC2 : CreateVerified AccessInstance
	ec2 : Fournisseur CreateVerified AccessTrust
	EC2 : CreateVolume
	EC2 : CreateVpc
	ec2 : Point de terminaison CreateVpc
	ec2 : Notification CreateVpc EndpointConnection
	ec2 : Configuration CreateVpc EndpointService
	EC2 : CreateVpc PeeringConnection
	ec2 : Connexion CreateVpn
	EC2 : CreateVpn ConnectionRoute
	ec2 : Passerelle CreateVpn
	ec2 : Passerelle DeleteCarrier
	EC2 : DeleteClient VpnEndpoint
	EC2 : DeleteClient VpnRoute
	ec2 : DeleteCoip Cidre
	ec2 : Piscine DeleteCoip
	ec2 : Passerelle DeleteCustomer
	ec2 : Options DeleteDhcp
	ec2 : Passerelle DeleteEgress OnlyInternet
	EC2 : DeleteFleets
	ec2 : DeleteFlow Journaux

Préfixe de service	Actions
	ec2 : Image DeleteFpga
	EC2 : DeleteInstance ConnectEndpoint
	EC2 : DeleteInstance EventWindow
	ec2 : Passerelle DeleteInternet
	EC2 : DeleteIpam
	ec2 : Piscine DeleteIpam
	EC2 : DeleteIpam ResourceDiscovery
	ec2 : DeleteIpam Champ d'application
	ec2 : paire DeleteKey
	ec2 : Modèle DeleteLaunch
	EC2 : DeleteLaunch TemplateVersions
	EC2 : DeleteLocal GatewayRoute
	ec2 : Tableau DeleteLocal GatewayRoute
	ec2 : Association DeleteLocal GatewayRoute TableVirtual Interface Group
	ec2 : Association DeleteLocal GatewayRoute TableVpc
	EC2 : DeleteManaged PrefixList
	ec2 : Passerelle DeleteNat
	ec2 : Acl DeleteNetwork
	EC2 : DeleteNetwork AclEntry
	ec2 : DeleteNetwork InsightsAccess Champ d'application

Préfixe de service	Actions
	EC2 : DeleteNetwork InsightsAccess ScopeAnalysis
	EC2 : DeleteNetwork InsightsAnalysis
	EC2 : DeleteNetwork InsightsPath
	ec2 : Interface DeleteNetwork
	EC2 : DeleteNetwork InterfacePermission
	ec2 : Groupe DeletePlacement
	ec2 : IPv4 Pool DeletePublic
	EC2 : DeleteQueued ReservedInstances
	EC2 : DeleteRoute
	ec2 : Tableau DeleteRoute
	ec2 : Groupe DeleteSecurity
	EC2 : DeleteSnapshot
	EC2 : DeleteSpot DatafeedSubscription
	EC2 : DeleteSubnet
	EC2 : DeleteSubnet CidrReservation
	EC2 : DeleteTraffic MirrorFilter
	ec2 : DeleteTraffic MirrorFilter Règle
	EC2 : DeleteTraffic MirrorSession
	EC2 : DeleteTraffic MirrorTarget
	ec2 : Passerelle DeleteTransit
	EC2 : DeleteTransit GatewayConnect

Préfixe de service	Actions
	ec2 : pair DeleteTransit GatewayConnect
	ec2 : Domaine DeleteTransit GatewayMulticast
	ec2 : Pièce jointe DeleteTransit GatewayPeering
	ec2 : Tableau DeleteTransit GatewayPolicy
	EC2 : DeleteTransit GatewayPrefix ListReference
	EC2 : DeleteTransit GatewayRoute
	ec2 : Tableau DeleteTransit GatewayRoute
	EC2 : DeleteTransit GatewayRoute TableAnnouncement
	ec2 : Pièce jointe DeleteTransit GatewayVpc
	EC2 : DeleteVerified AccessEndpoint
	EC2 : DeleteVerified AccessGroup
	EC2 : DeleteVerified AccessInstance
	ec2 : Fournisseur DeleteVerified AccessTrust
	EC2 : DeleteVolume
	EC2 : DeleteVpc
	ec2 : Notifications DeleteVpc EndpointConnection
	ec2 : Points de terminaison DeleteVpc
	ec2 : Configurations DeleteVpc EndpointService
	EC2 : DeleteVpc PeeringConnection
	ec2 : Connexion DeleteVpn
	EC2 : DeleteVpn ConnectionRoute

Préfixe de service	Actions
	ec2 : Passerelle DeleteVpn
	ec2 : DeprovisionByoip Cidre
	ec2 : Byosan DeprovisionIpam
	EC2 : DeprovisionIpam PoolCidr
	ec2 : IPv4 DeprovisionPublic PoolCidr
	EC2 : DeregisterImage
	ec2 : Attributs DeregisterInstance EventNotification
	EC2 : DeregisterTransit GatewayMulticast GroupMembers
	EC2 : DeregisterTransit GatewayMulticast GroupSources
	ec2 : Attributs DescribeAccount
	EC2 : DescribeAddresses
	ec2 : Attribut DescribeAddresses
	ec2 : Transferts DescribeAddress
	EC2 : DescribeAggregate IdFormat
	ec2 : Zones DescribeAvailability
	EC2 : DescribeAws NetworkPerformance MetricSubscriptions
	ec2 : Tâches DescribeBundle
	ec2 : DescribeByoip Cidres
	EC2 : DescribeCapacity ReservationFleets
	ec2 : Réservations DescribeCapacity
	ec2 : Passerelles DescribeCarrier

Préfixe de service	Actions
	EC2 : DescribeClassic LinkInstances
	ec2 : Règles DescribeClient VpnAuthorization
	EC2 : DescribeClient VpnConnections
	EC2 : DescribeClient VpnEndpoints
	EC2 : DescribeClient VpnRoutes
	ec2 : Réseaux DescribeClient VpnTarget
	ec2 : Piscines DescribeCoip
	ec2 : Tâches DescribeConversion
	ec2 : Passerelles DescribeCustomer
	ec2 : Options DescribeDhcp
	ec2 : Passerelles DescribeEgress OnlyInternet
	ec2 : GPU DescribeElastic
	EC2 : DescribeExport ImageTasks
	ec2 : Tâches DescribeExport
	EC2 : DescribeFast LaunchImages
	EC2 : DescribeFast SnapshotRestores
	ec2 : Historique DescribeFleet
	ec2 : Instances DescribeFleet
	EC2 : DescribeFleets
	ec2 : DescribeFlow Journaux
	EC2 : DescribeFpga ImageAttribute

Préfixe de service	Actions
	ec2 : Photos DescribeFpga
	EC2 : DescribeHost ReservationOfferings
	ec2 : Réservations DescribeHost
	EC2 : DescribeHosts
	ec2 : Associations DescribeIam InstanceProfile
	EC2 : DescribeIdentity IdFormat
	ec2 : Format DescribeId
	ec2 : Attribut DescribeImage
	EC2 : DescribeImages
	EC2 : DescribeImport ImageTasks
	EC2 : DescribeImport SnapshotTasks
	ec2 : Attribut DescribeInstance
	EC2 : DescribeInstance ConnectEndpoints
	EC2 : DescribeInstance CreditSpecifications
	ec2 : Attributs DescribeInstance EventNotification
	EC2 : DescribeInstance EventWindows
	EC2 : DescribeInstances
	ec2 : État DescribeInstance
	ec2 : Topologie DescribeInstance
	EC2 : DescribeInstance TypeOfferings
	ec2 : Types DescribeInstance

Préfixe de service	Actions
	ec2 : Passerelles DescribeInternet
	ec2 : Byosan DescribeIpam
	ec2 : Piscines DescribeIpam
	EC2 : DescribeIpam ResourceDiscoveries
	ec2 : Associations DescribeIpam ResourceDiscovery
	EC2 : DescribeIpams
	ec2 : DescribeIpam Éscopes
	ec2 : 6 piscines DescribeIpv
	ec2 : Paires DescribeKey
	ec2 : Modèles DescribeLaunch
	EC2 : DescribeLaunch TemplateVersions
	ec2 : Tableaux DescribeLocal GatewayRoute
	ec2 : Associations DescribeLocal GatewayRoute TableVirtual InterfaceGroup
	ec2 : Associations DescribeLocal GatewayRoute TableVpc
	ec2 : Passerelles DescribeLocal
	EC2 : DescribeLocal GatewayVirtual InterfaceGroups
	ec2 : Interfaces DescribeLocal GatewayVirtual
	ec2 : Instantanés DescribeLocked
	ec2 : Hôtes DescribeMac
	EC2 : DescribeManaged PrefixLists

Préfixe de service	Actions
	ec2 : Adresses DescribeMoving
	ec2 : Passerelles DescribeNat
	ec2 : ACL DescribeNetwork
	EC2 : DescribeNetwork InsightsAccess ScopeAnalyses
	ec2 : DescribeNetwork InsightsAccess Éscopes
	EC2 : DescribeNetwork InsightsAnalyses
	EC2 : DescribeNetwork InsightsPaths
	EC2 : DescribeNetwork InterfaceAttribute
	EC2 : DescribeNetwork InterfacePermissions
	ec2 : Interfaces DescribeNetwork
	ec2 : Groupes DescribePlacement
	ec2 : Listes DescribePrefix
	EC2 : DescribePrincipal IdFormat
	ec2 : IPv4 Pools DescribePublic
	EC2 : DescribeRegions
	ec2 : Tâches DescribeReplace RootVolume
	ec2 : Instances DescribeReserved
	EC2 : DescribeReserved InstancesListings
	EC2 : DescribeReserved InstancesModifications
	EC2 : DescribeReserved InstancesOfferings
	ec2 : Tableaux DescribeRoute

Préfixe de service	Actions
	EC2 : DescribeScheduled InstanceAvailability
	ec2 : Instances DescribeScheduled
	EC2 : DescribeSecurity GroupReferences
	EC2 : DescribeSecurity GroupRules
	ec2 : Groupes DescribeSecurity
	ec2 : Attribut DescribeSnapshot
	EC2 : DescribeSnapshots
	EC2 : DescribeSnapshot TierStatus
	EC2 : DescribeSpot DatafeedSubscription
	EC2 : DescribeSpot FleetInstances
	ec2 : Historique DescribeSpot FleetRequest
	EC2 : DescribeSpot FleetRequests
	EC2 : DescribeSpot InstanceRequests
	EC2 : DescribeSpot PriceHistory
	EC2 : DescribeStale SecurityGroups
	EC2 : DescribeStore ImageTasks
	EC2 : DescribeSubnets
	EC2 : DescribeTraffic MirrorFilters
	EC2 : DescribeTraffic MirrorSessions
	EC2 : DescribeTraffic MirrorTargets
	EC2 : DescribeTransit GatewayAttachments

Préfixe de service	Actions
	ec2 : pairs DescribeTransit GatewayConnect
	EC2 : DescribeTransit GatewayConnects
	ec2 : Domaines DescribeTransit GatewayMulticast
	ec2 : Pièces jointes DescribeTransit GatewayPeering
	ec2 : Tableaux DescribeTransit GatewayPolicy
	EC2 : DescribeTransit GatewayRoute TableAnnouncements
	ec2 : Tableaux DescribeTransit GatewayRoute
	ec2 : Passerelles DescribeTransit
	ec2 : Pièces jointes DescribeTransit GatewayVpc
	EC2 : DescribeTrunk InterfaceAssociations
	EC2 : DescribeVerified AccessEndpoints
	EC2 : DescribeVerified AccessGroups
	EC2 : DescribeVerified AccessInstance LoggingConfigurations
	EC2 : DescribeVerified AccessInstances
	ec2 : Fournisseurs DescribeVerified AccessTrust
	ec2 : Attribut DescribeVolume
	EC2 : DescribeVolumes
	ec2 : Modifications DescribeVolumes
	ec2 : État DescribeVolume
	ec2 : Attribut DescribeVpc
	EC2 : DescribeVpc ClassicLink

Préfixe de service	Actions
	EC2 : DescribeVpc ClassicLink DnsSupport
	ec2 : Notifications DescribeVpc EndpointConnection
	EC2 : DescribeVpc EndpointConnections
	ec2 : Points de terminaison DescribeVpc
	ec2 : Configurations DescribeVpc EndpointService
	ec2 : Autorisations DescribeVpc EndpointService
	EC2 : DescribeVpc EndpointServices
	EC2 : DescribeVpc PeeringConnections
	EC2 : DescribeVpcs
	ec2 : Connexions DescribeVpn
	ec2 : Passerelles DescribeVpn
	EC2 : DetachClassic LinkVpc
	ec2 : Passerelle DetachInternet
	ec2 : Interface DetachNetwork
	ec2 : Fournisseur DetachVerified AccessTrust
	EC2 : DetachVolume
	ec2 : Passerelle DetachVpn
	ec2 : Transfert DisableAddress
	EC2 : DisableAws NetworkPerformance MetricSubscription
	ec2 : Par défaut DisableEbs EncryptionBy
	ec2 : Lancement DisableFast

Préfixe de service	Actions
	EC2 : DisableFast SnapshotRestores
	EC2 : DisableImage
	ec2 : Accès DisableImage BlockPublic
	ec2 : Obsolète DisableImage
	EC2 : DisableImage DeregistrationProtection
	ec2 : Compte DisableIpam OrganizationAdmin
	EC2 : DisableSerial ConsoleAccess
	ec2 : Accès DisableSnapshot BlockPublic
	EC2 : DisableTransit GatewayRoute TablePropagation
	EC2 : DisableVgw RoutePropagation
	EC2 : DisableVpc ClassicLink
	EC2 : DisableVpc ClassicLink DnsSupport
	EC2 : DisassociateAddress
	ec2 : Réseau DisassociateClient VpnTarget
	ec2 : Rôle DisassociateEnclave CertificateIam
	EC2 : DisassociateIam InstanceProfile
	EC2 : DisassociateInstance EventWindow
	ec2 : Byosan DisassociateIpam
	EC2 : DisassociateIpam ResourceDiscovery
	EC2 : DisassociateNat GatewayAddress
	ec2 : Tableau DisassociateRoute

Préfixe de service	Actions
	EC2 : DisassociateSubnet CidrBlock
	ec2 : Domaine DisassociateTransit GatewayMulticast
	ec2 : Tableau DisassociateTransit GatewayPolicy
	ec2 : Tableau DisassociateTransit GatewayRoute
	ec2 : Interface DisassociateTrunk
	EC2 : DisassociateVpc CidrBlock
	ec2 : Transfert EnableAddress
	EC2 : EnableAws NetworkPerformance MetricSubscription
	ec2 : Par défaut EnableEbs EncryptionBy
	ec2 : Lancement EnableFast
	EC2 : EnableFast SnapshotRestores
	EC2 : EnableImage
	ec2 : Accès EnableImage BlockPublic
	ec2 : Obsolète EnableImage
	EC2 : EnableImage DeregistrationProtection
	ec2 : Compte EnableIpam OrganizationAdmin
	ec2 : Partage EnableReachability AnalyzerOrganization
	EC2 : EnableSerial ConsoleAccess
	ec2 : Accès EnableSnapshot BlockPublic
	EC2 : EnableTransit GatewayRoute TablePropagation
	EC2 : EnableVgw RoutePropagation

Préfixe de service	Actions
	ec2 : E/S EnableVolume
	EC2 : EnableVpc ClassicLink
	EC2 : EnableVpc ClassicLink DnsSupport
	ec2 : Liste ExportClient VpnClient CertificateRevocation
	ec2 : Configuration ExportClient VpnClient
	EC2 : ExportImage
	EC2 : ExportTransit GatewayRoutes
	EC2 : GetAssociated EnclaveCertificate IamRoles
	ec2 : IPv6 GetAssociated PoolCidrs
	ec2 : Données GetAws NetworkPerformance
	EC2 : GetCapacity ReservationUsage
	EC2 : GetCoip PoolUsage
	ec2 : Sortie GetConsole
	ec2 : GetConsole Capture d'écran
	EC2 : GetDefault CreditSpecification
	EC2 : GetEbs DefaultKms KeyId
	ec2 : Par défaut GetEbs EncryptionBy
	ec2 : Modèle GetFlow LogsIntegration
	ec2 : Réservation GetGroups ForCapacity
	ec2 : Aperçu GetHost ReservationPurchase
	EC2 : GetImage BlockPublic AccessState

Préfixe de service	Actions
	EC2 : GetInstance MetadataDefaults
	ec2 : Pub GetInstance TpmEk
	EC2 : GetInstance TypesFrom InstanceRequirements
	EC2 : GetInstance UefiData
	EC2 : GetIpam AddressHistory
	EC2 : GetIpam DiscoveredAccounts
	ec2 : Adresses GetIpam DiscoveredPublic
	ec2 : GetIpam DiscoveredResource Cidres
	EC2 : GetIpam PoolAllocations
	EC2 : GetIpam PoolCidrs
	EC2 : GetIpam ResourceCidrs
	EC2 : GetLaunch TemplateData
	ec2 : Associations GetManaged PrefixList
	ec2 : Entrées GetManaged PrefixList
	ec2 : Conclusions GetNetwork InsightsAccess ScopeAnalysis
	EC2 : GetNetwork InsightsAccess ScopeContent
	ec2 : Données GetPassword
	ec2 : GetReserved InstancesExchange Citation
	ec2 : PVC GetSecurity GroupsFor
	ec2 : État GetSerial ConsoleAccess
	EC2 : GetSnapshot BlockPublic AccessState

Préfixe de service	Actions
	EC2 : GetSpot PlacementScores
	EC2 : GetSubnet CidrReservations
	ec2 : Propagations GetTransit GatewayAttachment
	EC2 : GetTransit GatewayMulticast DomainAssociations
	EC2 : GetTransit GatewayPolicy TableAssociations
	EC2 : GetTransit GatewayPolicy TableEntries
	EC2 : GetTransit GatewayPrefix ListReferences
	EC2 : GetTransit GatewayRoute TableAssociations
	EC2 : GetTransit GatewayRoute TablePropagations
	ec2 : Politique GetVerified AccessEndpoint
	ec2 : Politique GetVerified AccessGroup
	EC2 : GetVpn ConnectionDevice SampleConfiguration
	ec2 : Types GetVpn ConnectionDevice
	ec2 : État GetVpn TunnelReplacement
	ec2 : Liste ImportClient VpnClient CertificateRevocation
	EC2 : ImportImage
	EC2 : ImportInstance
	ec2 : paire ImportKey
	EC2 : ImportSnapshot
	EC2 : ImportVolume
	ec2 : Poubelle ListImages InRecycle

Préfixe de service	Actions
	ec2 : Poubelle ListSnapshots InRecycle
	EC2 : LockSnapshot
	ec2 : Attribut ModifyAddress
	EC2 : ModifyAvailability ZoneGroup
	ec2 : Réservation ModifyCapacity
	EC2 : ModifyCapacity ReservationFleet
	EC2 : ModifyClient VpnEndpoint
	EC2 : ModifyDefault CreditSpecification
	EC2 : ModifyEbs DefaultKms KeyId
	EC2 : ModifyFleet
	EC2 : ModifyFpga ImageAttribute
	EC2 : ModifyHosts
	EC2 : ModifyIdentity IdFormat
	ec2 : Format ModifyId
	ec2 : Attribut ModifyImage
	ec2 : Attribut ModifyInstance
	ec2 : Attributs ModifyInstance CapacityReservation
	EC2 : ModifyInstance CreditSpecification
	ec2 : Heure ModifyInstance EventStart
	EC2 : ModifyInstance EventWindow
	EC2 : ModifyInstance MaintenanceOptions

Préfixe de service	Actions
	EC2 : ModifyInstance MetadataDefaults
	EC2 : ModifyInstance MetadataOptions
	ec2 : Placement ModifyInstance
	EC2 : ModifyIpam
	ec2 : Piscine ModifyIpam
	EC2 : ModifyIpam ResourceCidr
	EC2 : ModifyIpam ResourceDiscovery
	ec2 : ModifyIpam Champ d'application
	ec2 : Modèle ModifyLaunch
	EC2 : ModifyLocal GatewayRoute
	EC2 : ModifyManaged PrefixList
	EC2 : ModifyNetwork InterfaceAttribute
	ec2 : Options ModifyPrivate DnsName
	ec2 : Instances ModifyReserved
	EC2 : ModifySecurity GroupRules
	ec2 : Attribut ModifySnapshot
	ec2 : niveau ModifySnapshot
	EC2 : ModifySpot FleetRequest
	ec2 : Attribut ModifySubnet
	EC2 : ModifyTraffic MirrorFilter NetworkServices
	ec2 : ModifyTraffic MirrorFilter Règle

Préfixe de service	Actions
	EC2 : ModifyTraffic MirrorSession
	ec2 : Passerelle ModifyTransit
	EC2 : ModifyTransit GatewayPrefix ListReference
	ec2 : Pièce jointe ModifyTransit GatewayVpc
	EC2 : ModifyVerified AccessEndpoint
	ec2 : Politique ModifyVerified AccessEndpoint
	EC2 : ModifyVerified AccessGroup
	ec2 : Politique ModifyVerified AccessGroup
	EC2 : ModifyVerified AccessInstance
	EC2 : ModifyVerified AccessInstance LoggingConfiguration
	ec2 : Fournisseur ModifyVerified AccessTrust
	EC2 : ModifyVolume
	ec2 : Attribut ModifyVolume
	ec2 : Attribut ModifyVpc
	ec2 : Point de terminaison ModifyVpc
	ec2 : Notification ModifyVpc EndpointConnection
	ec2 : Configuration ModifyVpc EndpointService
	EC2 : ModifyVpc EndpointService PayerResponsibility
	ec2 : Autorisations ModifyVpc EndpointService
	ec2 : Options ModifyVpc PeeringConnection
	ec2 : Location ModifyVpc

Préfixe de service	Actions
	ec2 : Connexion ModifyVpn
	EC2 : ModifyVpn ConnectionOptions
	EC2 : ModifyVpn TunnelCertificate
	EC2 : ModifyVpn TunnelOptions
	EC2 : MonitorInstances
	EC2 : MoveAddress ToVpc
	ec2 : Ipam MoveByoip CidrTo
	ec2 : ProvisionByoip Cidre
	ec2 : Byosan ProvisionIpam
	EC2 : ProvisionIpam PoolCidr
	ec2 : IPv4 ProvisionPublic PoolCidr
	ec2 : Réservation PurchaseHost
	EC2 : PurchaseReserved InstancesOffering
	ec2 : Instances PurchaseScheduled
	EC2 : RebootInstances
	EC2 : RegisterImage
	ec2 : Attributs RegisterInstance EventNotification
	EC2 : RegisterTransit GatewayMulticast GroupMembers
	EC2 : RegisterTransit GatewayMulticast GroupSources
	EC2 : RejectTransit GatewayMulticast DomainAssociations
	ec2 : Pièce jointe RejectTransit GatewayPeering

Préfixe de service	Actions
	ec2 : Pièce jointe RejectTransit GatewayVpc
	EC2 : RejectVpc EndpointConnections
	EC2 : RejectVpc PeeringConnection
	EC2 : ReleaseAddress
	EC2 : ReleaseHosts
	EC2 : ReleaseIpam PoolAllocation
	ec2 : Association ReplaceIam InstanceProfile
	EC2 : ReplaceNetwork AclAssociation
	EC2 : ReplaceNetwork AclEntry
	EC2 : ReplaceRoute
	EC2 : ReplaceRoute TableAssociation
	EC2 : ReplaceTransit GatewayRoute
	ec2 : Tunnel ReplaceVpn
	ec2 : État ReportInstance
	ec2 : Flotte RequestSpot
	ec2 : Instances RequestSpot
	ec2 : Attribut ResetAddress
	EC2 : ResetEbs DefaultKms KeyId
	EC2 : ResetFpga ImageAttribute
	ec2 : Attribut ResetImage
	ec2 : Attribut ResetInstance

Préfixe de service	Actions
	EC2 : ResetNetwork InterfaceAttribute
	ec2 : Attribut ResetSnapshot
	EC2 : RestoreAddress ToClassic
	ec2 : Poubelle RestoreImage FromRecycle
	ec2 : Version RestoreManaged PrefixList
	ec2 : Poubelle RestoreSnapshot FromRecycle
	ec2 : niveau RestoreSnapshot
	EC2 : RevokeClient VpnIngress
	EC2 : RevokeSecurity GroupEgress
	EC2 : RevokeSecurity GroupIngress
	EC2 : RunInstances
	ec2 : Instances RunScheduled
	EC2 : SearchLocal GatewayRoutes
	ec2 : Groupes SearchTransit GatewayMulticast
	EC2 : SearchTransit GatewayRoutes
	ec2 : Interruption SendDiagnostic
	EC2 : StartInstances
	EC2 : StartNetwork InsightsAccess ScopeAnalysis
	EC2 : StartNetwork InsightsAnalysis
	ec2 : Vérification StartVpc EndpointService PrivateDns
	EC2 : StopInstances

Préfixe de service	Actions
	EC2 : TerminateClient VpnConnections
	EC2 : TerminateInstances
	ec2 : 6 Adresses UnassignIpv
	EC2 : UnassignPrivate IpAddresses
	ec2 : Adresse UnassignPrivate NatGateway
	EC2 : UnlockSnapshot
	EC2 : UnmonitorInstances
	EC2 : UpdateSecurity GroupRule DescriptionsEgress
	EC2 : UpdateSecurity GroupRule DescriptionsIngress
	ec2 : WithdrawByoip Cidre

Préfixe de service	Actions
ecr	ecr : BatchCheck LayerAvailability ecr : Image BatchDelete ecr : Image BatchGet ecr : Configuration BatchGet RepositoryScanning ecr : Télécharger CompleteLayer ecr : CreatePull ThroughCache Règle ecr : CreateRepository ecr : CreateRepository CreationTemplate ecr : Politique DeleteLifecycle ecr : DeletePull ThroughCache Règle ecr : Politique DeleteRegistry ecr : DeleteRepository ecr : DeleteRepository CreationTemplate ecr : Politique DeleteRepository ecr : DescribeImage ReplicationStatus ecr : DescribeImages ecr : DescribeImage ScanFindings ecr : Règles DescribePull ThroughCache ecr : DescribeRegistry ecr : DescribeRepositories ecr : GetAuthorization jeton

Préfixe de service	Actions
	ecr : couche GetDownload UrlFor ecr : Politique GetLifecycle ecr : GetLifecycle PolicyPreview ecr : Politique GetRegistry ecr : GetRegistry ScanningConfiguration ecr : Politique GetRepository ecr : Télécharger InitiateLayer ecr : ListImages ecr : PutImage ecr : PutImage ScanningConfiguration ecr : Politique PutRegistry ecr : PutRegistry ScanningConfiguration ecr : Configuration PutReplication ecr : Numériser StartImage ecr : StartLifecycle PolicyPreview ecr : UpdatePull ThroughCache Règle ecr : Pièce UploadLayer ecr : ValidatePull ThroughCache Règle

Préfixe de service	Actions
ecr-public	ecr-public : BatchCheck LayerAvailability
	ecr-public : Image BatchDelete
	ecr-public : Télécharger CompleteLayer
	ecr-public : CreateRepository
	ecr-public : DeleteRepository
	ecr-public : Politique DeleteRepository
	ecr-public : DescribeImages
	ecr-public : DescribeRegistries
	ecr-public : DescribeRepositories
	ecr-public : Jeton GetAuthorization
	ecr-public : GetRegistry CatalogData
	ecr-public : GetRepository CatalogData
	ecr-public : Politique GetRepository
	ecr-public : Télécharger InitiateLayer
	ecr-public : PutImage
	ecr-public : PutRegistry CatalogData
	ecr-public : PutRepository CatalogData
	ecr-public : Politique SetRepository
	ecr-public : Partie 1 UploadLayer

Préfixe de service	Actions
ecs	ecs : CreateCapacity Prestataire ecs : CreateCluster ecs : CreateService ecs : CreateTask Set ecs : DeleteAccount Réglage ecs : DeleteAttributes ecs : DeleteCapacity Prestataire ecs : DeleteCluster ecs : DeleteService ecs : DeleteTask Définitions ecs : DeleteTask Set ecs : DeregisterContainer Instance ecs : DeregisterTask Définition ecs : DescribeCapacity Fournisseurs ecs : DescribeClusters ecs : DescribeContainer Instances ecs : DescribeServices ecs : DescribeTask Définition ecs : DescribeTasks ecs : DescribeTask Ensembles ecs : DiscoverPoll point de terminaison

Préfixe de service	Actions
	ecs : ExecuteCommand ecs : GetTask Protection ecs : ListAccount Réglages ecs : ListAttributes ecs : ListClusters ecs : ListContainer Instances ecs : ListServices ecs : ListServices ByNamespace ecs : ListTask DefinitionFamilies ecs : ListTask Définitions ecs : ListTasks ecs : PutAccount Réglage ecs : PutAccount SettingDefault ecs : PutAttributes ecs : PutCluster CapacityProviders ecs : RegisterContainer Instance ecs : RegisterTask Définition ecs : RunTask ecs : StartTask ecs : StopTask ecs : SubmitAttachment StateChanges

Préfixe de service	Actions
	ecs : SubmitContainer StateChange ecs : SubmitTask StateChange ecs : UpdateCapacity Prestataire ecs : UpdateCluster ecs : UpdateCluster Réglages ecs : UpdateContainer Agent ecs : UpdateContainer InstancesState ecs : UpdateService ecs : UpdateService PrimaryTask Set ecs : UpdateTask Protection ecs : UpdateTask Set

Préfixe de service	Actions
eks	Reks : AssociateAccess Politique Eks : AssociateEncryption Config Eks : AssociateIdentity ProviderConfig Mots-clés : CreateAccess Entrée Eks : CreateAddon Eks : CreateCluster Eks : CreateEks AnywhereSubscription eks : CreateFargate Profil Eks : CreateNodegroup Mots-clés : DeleteAccess Entrée Eks : DeleteAddon Eks : DeleteCluster Eks : DeleteEks AnywhereSubscription eks : DeleteFargate Profil Eks : DeleteNodegroup Eks : DeletePod IdentityAssociation Eks : DeregisterCluster Mots-clés : DescribeAccess Entrée Eks : DescribeAddon Mots clés : DescribeAddon Configuration Mots clés : DescribeAddon Versions

Préfixe de service	Actions
	Eks : DescribeCluster
	Eks : DescribeEks AnywhereSubscription
	eks : DescribeFargate Profil
	Eks : DescribeIdentity ProviderConfig
	Eks : DescribeInsight
	Eks : DescribeNodegroup
	Eks : DescribePod IdentityAssociation
	Eks : DescribeUpdate
	Reks : DisassociateAccess Politique
	Eks : DisassociateIdentity ProviderConfig
	Reks : ListAccess Entrées
	Reks : ListAccess Politiques
	Eks : ListAddons
	Eks : ListAssociated AccessPolicies
	Eks : ListClusters
	Eks : ListEks AnywhereSubscriptions
	eks : ListFargate Profils
	Eks : ListIdentity ProviderConfigs
	Eks : ListInsights
	Eks : ListNodegroups
	Eks : ListPod IdentityAssociations

Préfixe de service	Actions
	Eks : ListUpdates Eks : RegisterCluster Mots-clés : UpdateAccess Entrée Eks : UpdateAddon Eks : UpdateCluster Config Mots clés : UpdateCluster Version Eks : UpdateEks AnywhereSubscription Eks : UpdateNodegroup Config Mots clés : UpdateNodegroup Version Eks : UpdatePod IdentityAssociation
elastic-inference	elastic-inference : Offres DescribeAccelerator inférence élastique : DescribeAccelerators Elastic-Inference : Types DescribeAccelerator

Préfixe de service	Actions
elasticache	elasticache : Ingress AuthorizeCache SecurityGroup
	Cache élastique : BatchApply UpdateAction
	Cache élastique : BatchStop UpdateAction
	Cache élastique : CompleteMigration
	Cache élastique : CopyServerless CacheSnapshot
	Cache élastique : CopySnapshot
	elasticache : Cluster CreateCache
	Cache élastique : CreateCache ParameterGroup
	Cache élastique : CreateCache SecurityGroup
	Cache élastique : CreateCache SubnetGroup
	Cache élastique : CreateGlobal ReplicationGroup
	elasticache : Groupe CreateReplication
	elasticache : cache CreateServerless
	Cache élastique : CreateServerless CacheSnapshot
	Cache élastique : CreateSnapshot
	Cache élastique : CreateUser
	elasticache : Groupe CreateUser
	elasticache : Groupe DecreaseNode GroupsIn GlobalReplication
	elasticache : Count DecreaseReplica
	elasticache : Cluster DeleteCache
	Cache élastique : DeleteCache ParameterGroup

Préfixe de service	Actions
	Cache élastique : DeleteCache SecurityGroup
	Cache élastique : DeleteCache SubnetGroup
	Cache élastique : DeleteGlobal ReplicationGroup
	elasticache : Groupe DeleteReplication
	elasticache : cache DeleteServerless
	Cache élastique : DeleteServerless CacheSnapshot
	Cache élastique : DeleteSnapshot
	Cache élastique : DeleteUser
	elasticache : Groupe DeleteUser
	elasticache : Clusters DescribeCache
	Cache élastique : DescribeCache EngineVersions
	Cache élastique : DescribeCache ParameterGroups
	elasticache : Paramètres DescribeCache
	Cache élastique : DescribeCache SecurityGroups
	Cache élastique : DescribeCache SubnetGroups
	Cache élastique : DescribeEngine DefaultParameters
	Cache élastique : DescribeEvents
	Cache élastique : DescribeGlobal ReplicationGroups
	elasticache : Groupes DescribeReplication
	Cache élastique : DescribeReserved CacheNodes
	elasticache : Offres DescribeReserved CacheNodes

Préfixe de service	Actions
	elasticache : Caches DescribeServerless
	Cache élastique : DescribeServerless CacheSnapshots
	elasticache : Mises à jour DescribeService
	Cache élastique : DescribeSnapshots
	elasticache : Actions DescribeUpdate
	elasticache : Groupes DescribeUser
	Cache élastique : DescribeUsers
	Cache élastique : DisassociateGlobal ReplicationGroup
	Cache élastique : ExportServerless CacheSnapshot
	Cache élastique : FailoverGlobal ReplicationGroup
	elasticache : Groupe IncreaseNode GroupsIn GlobalReplication
	elasticache : Count IncreaseReplica
	elasticache : Modifications ListAllowed NodeType
	elasticache : Cluster ModifyCache
	Cache élastique : ModifyCache ParameterGroup
	Cache élastique : ModifyCache SubnetGroup
	Cache élastique : ModifyGlobal ReplicationGroup
	elasticache : Groupe ModifyReplication
	elasticache : Configuration ModifyReplication GroupShard
	elasticache : cache ModifyServerless
	Cache élastique : ModifyUser

Préfixe de service	Actions
	elasticache : Groupe ModifyUser elasticache : Offre PurchaseReserved CacheNodes Cache élastique : RebalanceSlots InGlobal ReplicationGroup elasticache : Cluster RebootCache Cache élastique : ResetCache ParameterGroup elasticache : Ingress RevokeCache SecurityGroup Cache élastique : StartMigration Cache élastique : TestFailover Cache élastique : TestMigration

Préfixe de service	Actions
elasticbeanstalk	elasticbeanstalk : Mise à jour AbortEnvironment tige de haricot élastique : ApplyEnvironment ManagedAction tige de haricot élastique : AssociateEnvironment OperationsRole elasticbeanstalk:CheckDNSAvailability tige de haricot élastique : ComposeEnvironments tige de haricot élastique : CreateApplication elasticbeanstalk : Version CreateApplication elasticbeanstalk : Modèle CreateConfiguration tige de haricot élastique : CreateEnvironment elasticbeanstalk : Version CreatePlatform elasticbeanstalk : Emplacement CreateStorage tige de haricot élastique : DeleteApplication elasticbeanstalk : Version DeleteApplication elasticbeanstalk : Modèle DeleteConfiguration elasticbeanstalk : Configuration DeleteEnvironment elasticbeanstalk : Version DeletePlatform elasticbeanstalk : Attributs DescribeAccount tige de haricot élastique : DescribeApplications elasticbeanstalk : Versions DescribeApplication elasticbeanstalk : options DescribeConfiguration elasticbeanstalk : Paramètres DescribeConfiguration

Préfixe de service	Actions
	elasticbeanstalk : Health DescribeEnvironment elasticbeanstalk : Historique DescribeEnvironment ManagedAction tige de haricot élastique : DescribeEnvironment ManagedActions elasticbeanstalk : Ressources DescribeEnvironment tige de haricot élastique : DescribeEnvironments tige de haricot élastique : DescribeEvents elasticbeanstalk : Health DescribeInstances elasticbeanstalk : Version DescribePlatform tige de haricot élastique : DisassociateEnvironment OperationsRole tige de haricot élastique : ListAvailable SolutionStacks elasticbeanstalk : branches ListPlatform elasticbeanstalk : Versions ListPlatform tige de haricot élastique : RebuildEnvironment elasticbeanstalk : Informations RequestEnvironment elasticbeanstalk : Serveur RestartApp elasticbeanstalk : Informations RetrieveEnvironment elasticbeanstalk : CNames SwapEnvironment tige de haricot élastique : TerminateEnvironment tige de haricot élastique : UpdateApplication tige de haricot élastique : UpdateApplication ResourceLifecycle elasticbeanstalk : Version UpdateApplication

Préfixe de service	Actions
	elasticbeanstalk : Modèle UpdateConfiguration tige de haricot élastique : UpdateEnvironment elasticbeanstalk : Paramètres ValidateConfiguration

Préfixe de service	Actions
elasticfilesystem	système de fichiers élastique : Point CreateAccess
	elasticfilesystem : Système CreateFile
	elasticfilesystem : cible CreateMount
	elasticfilesystem : Configuration CreateReplication
	système de fichiers élastique : Point DeleteAccess
	elasticfilesystem : Système DeleteFile
	système de fichiers élastique : DeleteFile SystemPolicy
	elasticfilesystem : cible DeleteMount
	elasticfilesystem : Configuration DeleteReplication
	système de fichiers élastique : points DescribeAccess
	elasticfilesystem : Préférences DescribeAccount
	elasticfilesystem : Politique DescribeBackup
	système de fichiers élastique : DescribeFile SystemPolicy
	elasticfilesystem : Systèmes DescribeFile
	elasticfilesystem : Configuration DescribeLifecycle
	elasticfilesystem : cibles DescribeMount
	elasticfilesystem : Groupes DescribeMount TargetSecurity
	système de fichiers élastique : Configurations DescribeReplication
	elasticfilesystem : Groupes ModifyMount TargetSecurity
	elasticfilesystem : Préférences PutAccount
	elasticfilesystem : Politique PutBackup

Préfixe de service	Actions
	système de fichiers élastique : PutFile SystemPolicy elasticfilesystem : Configuration PutLifecycle elasticfilesystem : Système UpdateFile système de fichiers élastique : UpdateFile SystemProtection

Préfixe de service	Actions
elasticloadbalancing	elasticloadbalancing : Certificats AddListener équilibrage de charge élastique : AddTrust StoreRevocations équilibrage de charge élastique : ApplySecurity GroupsTo LoadBalancer elasticloadbalancing : sous-réseaux AttachLoad BalancerTo elasticloadbalancing : Vérifiez ConfigureHealth elasticloadbalancing : Politique CreateApp CookieStickiness ElasticLoad Balancing CookieStickiness : politique CreateLB équilibrage de charge élastique : CreateListener elasticloadbalancing : équilibreur CreateLoad équilibrage de charge élastique : CreateLoad BalancerListeners équilibrage de charge élastique : CreateLoad BalancerPolicy équilibrage de charge élastique : CreateRule elasticloadbalancing : Groupe CreateTarget elasticloadbalancing : Store CreateTrust équilibrage de charge élastique : DeleteListener elasticloadbalancing : équilibreur DeleteLoad équilibrage de charge élastique : DeleteLoad BalancerListeners équilibrage de charge élastique : DeleteLoad BalancerPolicy équilibrage de charge élastique : DeleteRule elasticloadbalancing : Groupe DeleteTarget

Préfixe de service	Actions
	elasticloadbalancing : Store DeleteTrust elasticloadbalancing : équilibreur DeregisterInstances FromLoad équilibrage de charge élastique : DeregisterTargets elasticloadbalancing : limites DescribeAccount Elasticloadbalancing : Health DescribeInstance elasticloadbalancing : Certificats DescribeListener équilibrage de charge élastique : DescribeListeners équilibrage de charge élastique : DescribeLoad BalancerAttributes équilibrage de charge élastique : DescribeLoad BalancerPolicies équilibrage de charge élastique : types DescribeLoad BalancerP olicy elasticloadbalancing : équilibreurs DescribeLoad équilibrage de charge élastique : DescribeRules elasticloadbalancing:DescribeSSLPolicies équilibrage de charge élastique : DescribeTarget GroupAttributes elasticloadbalancing : Groupes DescribeTarget Elasticloadbalancing : Health DescribeTarget équilibrage de charge élastique : DescribeTrust StoreAssociations équilibrage de charge élastique : DescribeTrust StoreRevocations Elasticloadbalancing : Boutiques DescribeTrust elasticloadbalancing : sous-réseaux DetachLoad BalancerFrom

Préfixe de service	Actions
	équilibrage de charge élastique : DisableAvailability ZonesFor LoadBalancer équilibrage de charge élastique : EnableAvailability ZonesFor LoadBalancer équilibrage de charge élastique : GetTrust StoreCa Certifica tesBundle elasticloadbalancing : Contenu GetTrust StoreRevocation équilibrage de charge élastique : ModifyListener équilibrage de charge élastique : ModifyLoad BalancerAttributes équilibrage de charge élastique : ModifyRule elasticloadbalancing : Groupe ModifyTarget équilibrage de charge élastique : ModifyTarget GroupAttributes elasticloadbalancing : Store ModifyTrust elasticloadbalancing : équilibreur RegisterInstances WithLoad équilibrage de charge élastique : RegisterTargets elasticloadbalancing : Certificats RemoveListener équilibrage de charge élastique : RemoveTrust StoreRevocations équilibrage de charge élastique : SetIp AddressType elasticloadbalancing : certificat SSL SetLoad BalancerListener elasticloadbalancing : serveur SetLoad BalancerPolicies ForBacken d équilibrage de charge élastique : SetLoad BalancerPolicies OfListener

Préfixe de service	Actions
	elasticloadbalancing : priorités SetRule elasticloadbalancing : Groupes SetSecurity équilibrage de charge élastique : SetSubnets
elastictranscoder	transcodeur élastique : CancelJob transcodeur élastique : CreateJob transcodeur élastique : CreatePipeline transcodeur élastique : CreatePreset transcodeur élastique : DeletePipeline transcodeur élastique : DeletePreset transcodeur élastique : ListJobs ByPipeline transcodeur élastique : ListJobs ByStatus transcodeur élastique : ListPipelines transcodeur élastique : ListPresets transcodeur élastique : ReadJob transcodeur élastique : ReadPipeline transcodeur élastique : ReadPreset transcodeur élastique : TestRole transcodeur élastique : UpdatePipeline elastictranscoder : Notifications UpdatePipeline elastictranscoder : État UpdatePipeline

Préfixe de service	Actions
emr-containers	emr-containers : Exécuter CancelJob emr-containers : Modèle CreateJob emr-containers : Point de terminaison CreateManaged emr-containers : Configuration CreateSecurity emr-containers : Cluster CreateVirtual emr-containers : Modèle DeleteJob emr-containers : Point de terminaison DeleteManaged emr-containers : Cluster DeleteVirtual emr-containers : Exécuter DescribeJob emr-containers : Modèle DescribeJob emr-containers : Point de terminaison DescribeManaged emr-containers : Configuration DescribeSecurity emr-containers : Cluster DescribeVirtual emr-containers : Informations d'identification GetManaged EndpointSession emr-containers : Exécute ListJob emr-containers : Modèles ListJob emr-containers : Points de terminaison ListManaged emr-containers : Configurations ListSecurity emr-containers : Clusters ListVirtual emr-containers : Exécuter StartJob

Préfixe de service	Actions
emr-serverless	emr-serverless : Exécuter CancelJob
	emr sans serveur : CreateApplication
	emr sans serveur : DeleteApplication
	emr sans serveur : GetApplication
	emr-serverless : Exécuter GetDashboard ForJob
	emr-serverless : Exécuter GetJob
	emr sans serveur : ListApplications
	emr-serverless : Exécute ListJob
	emr sans serveur : StartApplication
	emr-serverless : Exécuter StartJob
	emr sans serveur : StopApplication
	emr sans serveur : UpdateApplication

Préfixe de service	Actions
es	Oui : AcceptInbound Connexion Oui : AcceptInbound CrossCluster SearchConnection Oui : AssociatePackage Oui : AuthorizeVpc EndpointAccess Oui : CancelElasticsearch ServiceSoftware Mise à jour Oui : CancelService SoftwareUpdate Oui : CreateDomain Oui : CreateElasticsearch Domaine Oui : CreateOutbound Connexion Oui : CreateOutbound CrossCluster SearchConnection Oui : CreatePackage Oui : CreateVpc Endpoint Oui : DeleteDomain Oui : DeleteElasticsearch Domaine Oui : DeleteElasticsearch ServiceRole Oui : DeleteInbound Connexion Oui : DeleteInbound CrossCluster SearchConnection Oui : DeleteOutbound Connexion Oui : DeleteOutbound CrossCluster SearchConnection Oui : DeletePackage Oui : DeleteVpc Endpoint

Préfixe de service	Actions
	Oui : DescribeDomain Oui : DescribeDomain AutoTunes Oui : DescribeDomain ChangeProgress Oui : DescribeDomain Config Oui : DescribeDomain Health Oui : DescribeDomain Nœuds Oui : DescribeDomains Oui : DescribeDry RunProgress Oui : DescribeElasticsearch Domaine Oui : DescribeElasticsearch DomainConfig Oui : DescribeElasticsearch Domaines Oui : DescribeElasticsearch InstanceType Limites Oui : DescribeInbound Connexions Oui : DescribeInbound CrossCluster SearchConnections Oui : DescribeInstance TypeLimits Oui : DescribeOutbound Connexions Oui : DescribeOutbound CrossCluster SearchConnections Oui : DescribePackages Oui : DescribeReserved ElasticsearchInstance Offres Oui : DescribeReserved ElasticsearchInstances Oui : DescribeReserved InstanceOfferings

Préfixe de service	Actions
	Oui : DescribeReserved Instances Oui : Points de DescribeVpc terminaison Oui : DissociatePackage Oui : GetCompatible ElasticsearchVersions Oui : GetCompatible Versions Oui : GetData Source Oui : GetDomain MaintenanceStatus Oui : GetPackage VersionHistory Oui : GetUpgrade Histoire Oui : GetUpgrade État Oui : ListData Sources Oui : ListDomain Noms Oui : ListDomains ForPackage Oui : ListElasticsearch InstanceTypes Oui : ListElasticsearch Versions Oui : ListInstance TypeDetails Oui : ListPackages ForDomain Oui : ListScheduled Actions Oui : ListVersions Oui : ListVpc EndpointAccess Oui : Points de ListVpc terminaison

Préfixe de service	Actions
	Oui : ListVpc EndpointsFor Domaine Oui : PurchaseReserved ElasticsearchInstance Offre Oui : PurchaseReserved InstanceOffering Oui : RejectInbound Connexion Oui : RejectInbound CrossCluster SearchConnection Oui : RevokeVpc EndpointAccess Oui : StartDomain Entretien Oui : StartElasticsearch ServiceSoftware Mise à jour Oui : StartService SoftwareUpdate Oui : UpdateData Source Oui : UpdateDomain Config Oui : UpdateElasticsearch DomainConfig Oui : UpdatePackage Oui : UpdateScheduled Action Oui : UpdateVpc Endpoint Oui : UpgradeDomain Oui : UpgradeElasticsearch Domaine

Préfixe de service	Actions
événements	événements : ActivateEvent Source
	événements : CancelReplay
	événements : CreateApi Destination
	événements : CreateArchive
	événements : CreateConnection
	événements : CreateEndpoint
	événements : CreateEvent Bus
	événements : CreatePartner EventSource
	événements : DeactivateEvent Source
	événements : DeauthorizeConnection
	événements : DeleteApi Destination
	événements : DeleteArchive
	événements : DeleteConnection
	événements : DeleteEndpoint
	événements : DeleteEvent Bus
	événements : DeletePartner EventSource
	événements : DeleteRule
	événements : DescribeApi Destination
	événements : DescribeArchive
	événements : DescribeConnection
	événements : DescribeEndpoint

Préfixe de service	Actions
	événements : DescribeEvent Bus
	événements : DescribeEvent Source
	événements : DescribePartner EventSource
	événements : DescribeReplay
	événements : DescribeRule
	événements : DisableRule
	événements : EnableRule
	événements : ListApi Destinations
	événements : ListArchives
	événements : ListConnections
	événements : ListEndpoints
	événements : ListEvent Bus
	événements : ListEvent Sources
	événements : ListPartner EventSource Comptes
	événements : ListPartner EventSources
	événements : ListReplays
	événements : ListRule NamesBy Target
	événements : ListRules
	événements : ListTargets ByRule
	événements : PutPermission
	événements : PutRule

Préfixe de service	Actions
	événements : PutTargets
	événements : RemovePermission
	événements : RemoveTargets
	événements : StartReplay
	événements : TestEvent Pattern
	événements : UpdateApi Destination
	événements : UpdateArchive
	événements : UpdateConnection
	événements : UpdateEndpoint

Préfixe de service	Actions
evidently	évidemment : CreateExperiment évidemment : CreateFeature évidemment : CreateLaunch évidemment : CreateProject évidemment : CreateSegment évidemment : DeleteExperiment évidemment : DeleteFeature évidemment : DeleteLaunch évidemment : DeleteProject évidemment : DeleteSegment évidemment : GetExperiment évidemment : Résultats GetExperiment évidemment : GetFeature évidemment : GetLaunch évidemment : GetProject évidemment : GetSegment évidemment : ListExperiments évidemment : ListFeatures évidemment : ListLaunches évidemment : ListProjects évidemment : Références ListSegment

Préfixe de service	Actions
	évidemment : ListSegments évidemment : StartExperiment évidemment : StartLaunch évidemment : StopExperiment évidemment : StopLaunch évidemment : Motif TestSegment évidemment : UpdateExperiment évidemment : UpdateFeature évidemment : UpdateLaunch évidemment : UpdateProject évidemment : UpdateProject DataDelivery

Préfixe de service	Actions
finspace	espace fin : CreateEnvironment finspace : CreateKx Ensemble de modifications finspace : Cluster CreateKx finspace : CreateKx Base de données finspace : CreateKx Vue des données finspace : Environnement CreateKx espace fin : CreateKx ScalingGroup finspace : utilisateur CreateKx finspace : Volume CreateKx espace fin : CreateUser espace fin : DeleteEnvironment finspace : Cluster DeleteKx espace fin : DeleteKx ClusterNode finspace : DeleteKx Base de données finspace : DeleteKx Vue des données finspace : Environnement DeleteKx espace fin : DeleteKx ScalingGroup finspace : utilisateur DeleteKx finspace : Volume DeleteKx espace fin : GetEnvironment finspace : GetKx Ensemble de modifications

Préfixe de service	Actions
	finspace : Cluster GetKx
	espace fin : GetKx ConnectionString
	finspace : GetKx Base de données
	finspace : GetKx Vue des données
	finspace : Environnement GetKx
	espace fin : GetKx ScalingGroup
	finspace : utilisateur GetKx
	finspace : Volume GetKx
	finspace : État GetLoad SampleData SetGroup IntoEnvironment
	espace fin : GetUser
	espace fin : ListEnvironments
	finspace : ListKx Ensembles de modifications
	espace fin : ListKx ClusterNodes
	finspace : Clusters ListKx
	finspace : ListKx Bases de données
	finspace : ListKx Vues des données
	finspace : Environnements ListKx
	espace fin : ListKx ScalingGroups
	finspace : Utilisateurs ListKx
	finspace : Volumes ListKx
	espace fin : ListUsers

Préfixe de service	Actions
	finspace : Environnement LoadSample DataSet GroupInto finspace : ResetUser Mot de passe espace fin : UpdateEnvironment finspace : Configuration UpdateKx ClusterCode espace fin : UpdateKx ClusterDatabases finspace : UpdateKx Base de données finspace : UpdateKx Vue des données finspace : Environnement UpdateKx espace fin : UpdateKx EnvironmentNetwork finspace : utilisateur UpdateKx finspace : Volume UpdateKx espace fin : UpdateUser
firehose	Firehose : Stream CreateDelivery Firehose : Stream DeleteDelivery Firehose : Stream DescribeDelivery Firehose : Streams ListDelivery lance à incendie : StartDelivery StreamEncryption lance à incendie : StopDelivery StreamEncryption lance à incendie : UpdateDestination

Préfixe de service	Actions
fis	Fis : Modèle CreateExperiment poisson : CreateTarget AccountConfiguration Fis : Modèle DeleteExperiment poisson : DeleteTarget AccountConfiguration poisson : GetAction poisson : GetExperiment solution : Configuration GetExperiment TargetAccount Fis : Modèle GetExperiment poisson : GetTarget AccountConfiguration poisson : GetTarget ResourceType poisson : ListActions poisson : ListExperiment ResolvedTargets poisson : ListExperiments solution : Configurations ListExperiment TargetAccount solution : Modèles ListExperiment poisson : ListTarget AccountConfigurations poisson : ListTarget ResourceTypes poisson : StartExperiment poisson : StopExperiment Fis : Modèle UpdateExperiment poisson : UpdateTarget AccountConfiguration

Préfixe de service	Actions
fms	fms : Compte AssociateAdmin fms : AssociateThird PartyFirewall fms : Ressource BatchAssociate fms : Ressource BatchDisassociate fms : Liste DeleteApps fms : Canal DeleteNotification fms : DeletePolicy fms : Liste DeleteProtocols fms : Set DeleteResource fms : Compte DisassociateAdmin fms : DisassociateThird PartyFirewall fms : Compte GetAdmin fms : GetAdmin Champ d'application fms : Liste GetApps fms : Détail GetCompliance fms : Canal GetNotification fms : GetPolicy fms : État GetProtection fms : Liste GetProtocols fms : Set GetResource fms : GetThird PartyFirewall AssociationStatus

Préfixe de service	Actions
	fms : Détails GetViolation fms : Organisation ListAdmin AccountsFor fms : ListAdmins ManagingAccount fms : Listes ListApps fms : État ListCompliance fms : Ressources ListDiscovered fms : Comptes ListMember fms : ListPolicies fms : Listes ListProtocols fms : ListResource SetResources fms : Ensembles ListResource fms : ListThird PartyFirewall FirewallPolicies fms : Compte PutAdmin fms : Liste PutApps fms : Canal PutNotification fms : PutPolicy fms : Liste PutProtocols fms : Set PutResource

Préfixe de service	Actions
frauddetector	détecteur de fraude : variable BatchCreate
	détecteur de fraude : variable BatchGet
	détecteur de fraude : CancelBatch ImportJob
	détecteur de fraude : CancelBatch PredictionJob
	détecteur de fraude : CreateBatch ImportJob
	détecteur de fraude : CreateBatch PredictionJob
	détecteur de fraude : Version CreateDetector
	détecteur de fraude : CreateList
	détecteur de fraude : CreateModel
	détecteur de fraude : Version CreateModel
	détecteur de fraude : CreateRule
	détecteur de fraude : CreateVariable
	détecteur de fraude : DeleteBatch ImportJob
	détecteur de fraude : DeleteBatch PredictionJob
	détecteur de fraude : DeleteDetector
	détecteur de fraude : Version DeleteDetector
	détecteur de fraude : Type DeleteEntity
	détecteur de fraude : DeleteEvent
	détecteur de fraude : Type DeleteEvents ByEvent
	détecteur de fraude : Type DeleteEvent
	détecteur de fraude : modèle DeleteExternal

Préfixe de service	Actions
	détecteur de fraude : DeleteLabel
	détecteur de fraude : DeleteList
	détecteur de fraude : DeleteModel
	détecteur de fraude : Version DeleteModel
	détecteur de fraude : DeleteOutcome
	détecteur de fraude : DeleteRule
	détecteur de fraude : DeleteVariable
	détecteur de fraude : DescribeDetector
	détecteur de fraude : Versions DescribeModel
	détecteur de fraude : GetBatch ImportJobs
	détecteur de fraude : GetBatch PredictionJobs
	détecteur de fraude : État GetDelete EventsBy EventType
	détecteur de fraude : GetDetectors
	détecteur de fraude : Version GetDetector
	détecteur de fraude : types GetEntity
	détecteur de fraude : GetEvent
	détecteur de fraude : Prédiction GetEvent
	détecteur de fraude : GetEvent PredictionMetadata
	détecteur de fraude : types GetEvent
	détecteur de fraude : Modèles GetExternal
	Détecteur de fraude : GetKMS EncryptionKey

Préfixe de service	Actions
	détecteur de fraude : GetLabels
	détecteur de fraude : éléments GetList
	frauddetector : métadonnées GetLists
	détecteur de fraude : GetModels
	détecteur de fraude : Version GetModel
	détecteur de fraude : GetOutcomes
	détecteur de fraude : GetRules
	détecteur de fraude : GetVariables
	frauddetector : prédictions ListEvent
	détecteur de fraude : PutDetector
	détecteur de fraude : Type PutEntity
	détecteur de fraude : Type PutEvent
	détecteur de fraude : modèle PutExternal
	Détecteur de fraude : PUTKMS EncryptionKey
	détecteur de fraude : PutLabel
	détecteur de fraude : PutOutcome
	détecteur de fraude : SendEvent
	détecteur de fraude : Version UpdateDetector
	détecteur de fraude : UpdateDetector VersionMetadata
	détecteur de fraude : UpdateDetector VersionStatus
	détecteur de fraude : étiquette UpdateEvent

Préfixe de service	Actions
	détecteur de fraude : UpdateList
	détecteur de fraude : UpdateModel
	détecteur de fraude : Version UpdateModel
	détecteur de fraude : UpdateModel VersionStatus
	frauddetector : métadonnées UpdateRule
	détecteur de fraude : Version UpdateRule
	détecteur de fraude : UpdateVariable

Préfixe de service	Actions
fsx	télécopie : AssociateFile SystemAliases télécopie : CancelData RepositoryTask télécopie : CopyBackup télécopie : CreateData RepositoryTask fax : Cache CreateFile fsx : Système CreateFile fax : Backup CreateFile SystemFrom télécopie : CreateSnapshot télécopie : CreateStorage VirtualMachine télécopie : CreateVolume télécopie : CreateVolume FromBackup télécopie : DeleteBackup fax : Cache DeleteFile fsx : Système DeleteFile télécopie : DeleteSnapshot télécopie : DeleteStorage VirtualMachine télécopie : DeleteVolume télécopie : DescribeBackups télécopie : DescribeData RepositoryAssociations télécopie : DescribeData RepositoryTasks fax : Caches DescribeFile

Préfixe de service	Actions
	télécopie : DescribeFile SystemAliases fsx : Systèmes DescribeFile télécopie : DescribeShared VpcConfiguration télécopie : DescribeSnapshots télécopie : DescribeStorage VirtualMachines télécopie : DescribeVolumes télécopie : DisassociateFile SystemAliases fax : V3Locks ReleaseFile SystemNfs télécopie : RestoreVolume FromSnapshot télécopie : StartMisconfigured StateRecovery télécopie : UpdateData RepositoryAssociation fax : Cache UpdateFile fsx : Système UpdateFile télécopie : UpdateShared VpcConfiguration télécopie : UpdateSnapshot télécopie : UpdateStorage VirtualMachine télécopie : UpdateVolume

Préfixe de service	Actions
gamelift	gamelift : AcceptMatch
	gamelift : Serveur ClaimGame
	gamelift : CreateAlias
	gamelift : CreateBuild
	gamelift : CreateContainer GroupDefinition
	gamelift : CreateFleet
	gamelift : Localisations CreateFleet
	gamelift : CreateGame ServerGroup
	gamelift : Session CreateGame
	gamelift : CreateGame SessionQueue
	gamelift : CreateLocation
	gamelift : Configuration CreateMatchmaking
	gamelift : CreateMatchmaking RuleSet
	gamelift : Session CreatePlayer
	gamelift : Sessions CreatePlayer
	gamelift : CreateScript
	gamelift : CreateVpc PeeringAuthorization
	gamelift : CreateVpc PeeringConnection
	gamelift : DeleteAlias
	gamelift : DeleteBuild
	gamelift : DeleteContainer GroupDefinition

Préfixe de service	Actions
	gamelift : DeleteFleet
	gamelift : Localisations DeleteFleet
	gamelift : DeleteGame ServerGroup
	gamelift : DeleteGame SessionQueue
	gamelift : DeleteLocation
	gamelift : Configuration DeleteMatchmaking
	gamelift : DeleteMatchmaking RuleSet
	gamelift : Politique DeleteScaling
	gamelift : DeleteScript
	gamelift : DeleteVpc PeeringAuthorization
	gamelift : DeleteVpc PeeringConnection
	gamelift : DeregisterCompute
	gamelift : Serveur DeregisterGame
	gamelift : DescribeAlias
	gamelift : DescribeBuild
	gamelift : DescribeCompute
	gamelift : DescribeContainer GroupDefinition
	GameLift : Décrivez EC2 InstanceLimits
	gamelift : Attributs DescribeFleet
	gamelift : Capacité DescribeFleet
	gamelift : Événements DescribeFleet

Préfixe de service	Actions
	gamelift : DescribeFleet LocationAttributes
	gamelift : DescribeFleet LocationCapacity
	gamelift : DescribeFleet LocationUtilization
	gamelift : DescribeFleet PortSettings
	gamelift : Utilisation DescribeFleet
	gamelift : Serveur DescribeGame
	gamelift : DescribeGame ServerGroup
	gamelift : DescribeGame ServerInstances
	gamelift : DescribeGame SessionDetails
	gamelift : DescribeGame SessionPlacement
	gamelift : DescribeGame SessionQueues
	gamelift : Sessions DescribeGame
	gamelift : DescribeInstances
	gamelift : DescribeMatchmaking
	gamelift : Configurations DescribeMatchmaking
	gamelift : DescribeMatchmaking RuleSets
	gamelift : Sessions DescribePlayer
	gamelift : Configuration DescribeRuntime
	gamelift : Politiques DescribeScaling
	gamelift : DescribeScript
	gamelift : DescribeVpc PeeringAuthorizations

Préfixe de service	Actions
	gamelift : DescribeVpc PeeringConnections
	gamelift : Accès GetCompute
	gamelift : GetCompute AuthToken
	gamelift : URL GetGame SessionLog
	gamelift : Accès GetInstance
	gamelift : ListAliases
	gamelift : ListBuilds
	gamelift : ListCompute
	gamelift : ListContainer GroupDefinitions
	gamelift : ListFleets
	gamelift : ListGame ServerGroups
	gamelift : Serveurs ListGame
	gamelift : ListLocations
	gamelift : ListScripts
	gamelift : Politique PutScaling
	gamelift : RegisterCompute
	gamelift : Serveur RegisterGame
	gamelift : Informations d'identification RequestUpload
	gamelift : ResolveAlias
	gamelift : ResumeGame ServerGroup
	gamelift : Sessions SearchGame

Préfixe de service	Actions
	gamelift : Actions StartFleet
	gamelift : StartGame SessionPlacement
	gamelift : Backfill StartMatch
	gamelift : StartMatchmaking
	gamelift : Actions StopFleet
	gamelift : StopGame SessionPlacement
	gamelift : StopMatchmaking
	gamelift : SuspendGame ServerGroup
	gamelift : UpdateAlias
	gamelift : UpdateBuild
	gamelift : Attributs UpdateFleet
	gamelift : Capacité UpdateFleet
	gamelift : UpdateFleet PortSettings
	gamelift : Serveur UpdateGame
	gamelift : UpdateGame ServerGroup
	gamelift : Session UpdateGame
	gamelift : UpdateGame SessionQueue
	gamelift : Configuration UpdateMatchmaking
	gamelift : Configuration UpdateRuntime
	gamelift : UpdateScript
	gamelift : ValidateMatchmaking RuleSet

Préfixe de service	Actions
geo	géo : Consommateur AssociateTracker geo : Histoire BatchDelete DevicePosition géo : BatchDelete Geofence géo : BatchEvaluate Geofences géo : BatchGet DevicePosition géo : BatchPut Geofence géo : BatchUpdate DevicePosition géo : CalculateRoute géo : Matrix CalculateRoute géo : Collection CreateGeofence géo : CreateMap geo : Index CreatePlace geo : Calculatrice CreateRoute géo : CreateTracker géo : Collection DeleteGeofence géo : DeleteKey géo : DeleteMap geo : Index DeletePlace geo : Calculatrice DeleteRoute géo : DeleteTracker géo : Collection DescribeGeofence

Préfixe de service	Actions
	géo : DescribeKey géo : DescribeMap geo : Index DescribePlace geo : Calculatrice DescribeRoute géo : DescribeTracker géo : Consommateur DisassociateTracker géo : Position GetDevice géo : GetDevice PositionHistory géo : GetGeofence geo : GetMap Glyphes géo : Sprites GetMap géo : GetMap StyleDescriptor géo : Tuile GetMap géo : GetPlace géo : Positions ListDevice geo : Collections ListGeofence géo : ListGeofences géo : ListKeys géo : ListMaps geo : Indices ListPlace geo : Calculateurs ListRoute

Préfixe de service	Actions
	geo : Consommateurs ListTracker géo : ListTrackers géo : PutGeofence géo : Position SearchPlace IndexFor geo : Suggestions SearchPlace IndexFor geo : Texte SearchPlace IndexFor géo : Collection UpdateGeofence géo : UpdateKey géo : UpdateMap geo : Index UpdatePlace geo : Calculatrice UpdateRoute géo : UpdateTracker

Préfixe de service	Actions
glacier	glacier : AbortMultipart Télécharger
	glacier : AbortVault Écluse
	glacier : CompleteMultipart Télécharger
	glacier : CompleteVault Écluse
	glacier : CreateVault
	glacier : DeleteArchive
	glacier : DeleteVault
	glacier : DeleteVault AccessPolicy
	glacier : DeleteVault Notifications
	glacier : DescribeJob
	glacier : DescribeVault
	glacier : GetData RetrievalPolicy
	glacier : GetJob sortie
	glacier : GetVault AccessPolicy
	glacier : GetVault Écluse
	glacier : GetVault Notifications
	glacier : InitiateJob
	glacier : InitiateMultipart Télécharger
	glacier : InitiateVault Écluse
	glacier : ListJobs
	glacier : ListMultipart Téléchargements

Préfixe de service	Actions
	glacier : ListParts glacier : ListProvisioned Capacité glacier : ListVaults glacier : PurchaseProvisioned Capacité glacier : SetData RetrievalPolicy glacier : SetVault AccessPolicy glacier : SetVault Notifications glacier : UploadArchive glacier : UploadMultipart partie

Préfixe de service	Actions
grafana	Grafana : AssociateLicense Grafana : CreateWorkspace Grafana : CreateWorkspace ApiKey Grafana : DeleteWorkspace Grafana : DeleteWorkspace ApiKey Grafana : DescribeWorkspace grafana : Authentification DescribeWorkspace grafana : Configuration DescribeWorkspace Grafana : DisassociateLicense Grafana : ListPermissions Grafana : ListVersions Grafana : ListWorkspaces Grafana : UpdatePermissions Grafana : UpdateWorkspace grafana : Authentification UpdateWorkspace grafana : Configuration UpdateWorkspace

Préfixe de service	Actions
greengrass	herbe verte : AssociateRole ToGroup greengrass : Compte AssociateService RoleTo greengrass : Appareil BatchAssociate ClientDevice WithCore greengrass : Appareil BatchDisassociate ClientDevice FromCore herbe verte : CancelDeployment greengrass : Version CreateComponent greengrass : définition CreateConnector herbe verte : CreateConnector DefinitionVersion greengrass : définition CreateCore herbe verte : CreateCore DefinitionVersion herbe verte : CreateDeployment greengrass : définition CreateDevice herbe verte : CreateDevice DefinitionVersion greengrass : définition CreateFunction herbe verte : CreateFunction DefinitionVersion herbe verte : CreateGroup herbe verte : CreateGroup CertificateAuthority greengrass : Version CreateGroup greengrass : définition CreateLogger herbe verte : CreateLogger DefinitionVersion greengrass : définition CreateResource

Préfixe de service	Actions
	herbe verte : CreateResource DefinitionVersion
	herbe verte : CreateSoftware UpdateJob
	greengrass : définition CreateSubscription
	herbe verte : CreateSubscription DefinitionVersion
	herbe verte : DeleteComponent
	greengrass : définition DeleteConnector
	greengrass : définition DeleteCore
	greengrass : Appareil DeleteCore
	herbe verte : DeleteDeployment
	greengrass : définition DeleteDevice
	greengrass : définition DeleteFunction
	herbe verte : DeleteGroup
	greengrass : définition DeleteLogger
	greengrass : définition DeleteResource
	greengrass : définition DeleteSubscription
	herbe verte : DescribeComponent
	herbe verte : DisassociateRole FromGroup
	greengrass : Compte DisassociateService RoleFrom
	greengrass : Rôle GetAssociated
	herbe verte : GetBulk DeploymentStatus
	herbe verte : GetComponent

Préfixe de service	Actions
	herbe verte : GetComponent VersionArtifact
	greengrass : Informations GetConnectivity
	greengrass : définition GetConnector
	herbe verte : GetConnector DefinitionVersion
	greengrass : définition GetCore
	herbe verte : GetCore DefinitionVersion
	greengrass : Appareil GetCore
	herbe verte : GetDeployment
	greengrass : Statut GetDeployment
	greengrass : définition GetDevice
	herbe verte : GetDevice DefinitionVersion
	greengrass : définition GetFunction
	herbe verte : GetFunction DefinitionVersion
	herbe verte : GetGroup
	herbe verte : GetGroup CertificateAuthority
	herbe verte : GetGroup CertificateConfiguration
	greengrass : Version GetGroup
	greengrass : définition GetLogger
	herbe verte : GetLogger DefinitionVersion
	greengrass : définition GetResource
	herbe verte : GetResource DefinitionVersion

Préfixe de service	Actions
	greengrass : Compte GetService RoleFor
	greengrass : définition GetSubscription
	herbe verte : GetSubscription DefinitionVersion
	herbe verte : GetThing RuntimeConfiguration
	greengrass : Rapports ListBulk DeploymentDetailed
	greengrass : Déploiements ListBulk
	greengrass : Appareil ListClient DevicesAssociated WithCore
	herbe verte : ListComponents
	greengrass : ListComponent Versions
	greengrass : Définitions ListConnector
	herbe verte : ListConnector DefinitionVersions
	greengrass : Définitions ListCore
	herbe verte : ListCore DefinitionVersions
	greengrass : Appareils ListCore
	herbe verte : ListDeployments
	greengrass : Définitions ListDevice
	herbe verte : ListDevice DefinitionVersions
	greengrass : Déploiements ListEffective
	greengrass : Définitions ListFunction
	herbe verte : ListFunction DefinitionVersions
	herbe verte : ListGroup CertificateAuthorities

Préfixe de service	Actions
	herbe verte : ListGroups
	greengrass : ListGroup Versions
	greengrass : Composants ListInstalled
	greengrass : Définitions ListLogger
	herbe verte : ListLogger DefinitionVersions
	greengrass : Définitions ListResource
	herbe verte : ListResource DefinitionVersions
	greengrass : Définitions ListSubscription
	herbe verte : ListSubscription DefinitionVersions
	herbe verte : ResetDeployments
	greengrass : Déploiement StartBulk
	greengrass : Déploiement StopBulk
	greengrass : Informations UpdateConnectivity
	greengrass : définition UpdateConnector
	greengrass : définition UpdateCore
	greengrass : définition UpdateDevice
	greengrass : définition UpdateFunction
	herbe verte : UpdateGroup
	herbe verte : UpdateGroup CertificateConfiguration
	greengrass : définition UpdateLogger
	greengrass : définition UpdateResource

Préfixe de service	Actions
	greengrass : définition UpdateSubscription herbe verte : UpdateThing RuntimeConfiguration

Préfixe de service	Actions
groundstation	station au sol : CancelContact station au sol : CreateConfig station au sol : CreateDataflow EndpointGroup station au sol : CreateEphemeris groundstation : Profil CreateMission station au sol : DeleteConfig station au sol : DeleteDataflow EndpointGroup station au sol : DeleteEphemeris groundstation : Profil DeleteMission station au sol : DescribeContact station au sol : DescribeEphemeris station au sol : GetConfig station au sol : GetDataflow EndpointGroup station au sol : utilisation GetMinute groundstation : Profil GetMission station au sol : GetSatellite station au sol : ListConfigs station au sol : ListContacts station au sol : ListDataflow EndpointGroups station au sol : ListEphemerides station au sol : Stations ListGround

Préfixe de service	Actions
	groundstation : Profilés ListMission station au sol : ListSatellites station au sol : RegisterAgent station au sol : ReserveContact station au sol : État UpdateAgent station au sol : UpdateConfig station au sol : UpdateEphemeris groundstation : Profil UpdateMission

Préfixe de service	Actions
guardduty	guardduty : Invitation AcceptAdministrator devoir de garde : AcceptInvitation devoir de garde : ArchiveFindings devoir de garde : CreateDetector devoir de garde : CreateFilter guardduty:CreateIPSet devoir de garde : CreateMembers service de garde : Destination CreatePublishing guardduty : Conclusions CreateSample devoir de garde : CreateThreat IntelSet devoir de garde : DeclineInvitations devoir de garde : DeleteDetector devoir de garde : DeleteFilter devoir de garde : DeleteInvitations guardduty:DeleteIPSet devoir de garde : DeleteMembers service de garde : Destination DeletePublishing devoir de garde : DeleteThreat IntelSet guardduty : Scans DescribeMalware guardduty : Configuration DescribeOrganization service de garde : Destination DescribePublishing

Préfixe de service	Actions
	devoir de garde : DisableOrganization AdminAccount
	devoir de garde : DisassociateFrom AdministratorAccount
	devoir de garde : DisassociateFrom MasterAccount
	devoir de garde : DisassociateMembers
	devoir de garde : EnableOrganization AdminAccount
	guardduty : Compte GetAdministrator
	guardduty : Statistiques GetCoverage
	devoir de garde : GetDetector
	devoir de garde : GetFilter
	devoir de garde : GetFindings
	guardduty : Statistiques GetFindings
	devoir de garde : comte GetInvitations
	guardduty:GetIPSet
	devoir de garde : GetMalware ScanSettings
	guardduty : Compte GetMaster
	guardduty : Détecteurs GetMember
	devoir de garde : GetMembers
	guardduty : Statistiques GetOrganization
	service de garde : 7 jours GetRemaining FreeTrial
	devoir de garde : GetThreat IntelSet
	guardduty : Statistiques GetUsage

Préfixe de service	Actions
	devoir de garde : InviteMembers
	devoir de garde : ListCoverage
	devoir de garde : ListDetectors
	devoir de garde : ListFilters
	devoir de garde : ListFindings
	devoir de garde : ListInvitations
	guardduty:ListIPSets
	devoir de garde : ListMembers
	devoir de garde : ListOrganization AdminAccounts
	service de garde : Destinations ListPublishing
	devoir de garde : ListThreat IntelSets
	guardduty : Télémétrie SendSecurity
	guardduty : Scanner StartMalware
	guardduty : Membres StartMonitoring
	guardduty : Membres StopMonitoring
	devoir de garde : UnarchiveFindings
	devoir de garde : UpdateDetector
	devoir de garde : UpdateFilter
	guardduty : Commentaires UpdateFindings
	guardduty:UpdateIPSet
	devoir de garde : UpdateMalware ScanSettings

Préfixe de service	Actions
	guardduty : Détecteurs UpdateMember guardduty : Configuration UpdateOrganization service de garde : Destination UpdatePublishing devoir de garde : UpdateThreat IntelSet
healthlake	healthlake:CreateFHIRDatastore lac de santé : CreateResource healthlake:DeleteFHIRDatastore lac de santé : DeleteResource healthlake:DescribeFHIRDatastore Lac de santé : décrivez le FHIR ExportJob Lac de santé : décrivez le FHIR ImportJob lac de santé : GetCapabilities healthlake:ListFHIRDatastores Lac de santé : liste FHIR ExportJobs Lac de santé : liste FHIR ImportJobs lac de santé : ReadResource healthlake : Obtenez SearchWith healthlake : Poster SearchWith Lac de santé : StartFHIR ExportJob Lac de santé : StartFHIR ImportJob lac de santé : UpdateResource

Préfixe de service	Actions
honeycode	code d'identification : BatchCreate TableRows code d'identification : BatchDelete TableRows code d'identification : BatchUpdate TableRows code d'identification : BatchUpsert TableRows honeycode : Job DescribeTable DataImport honeycode : Données GetScreen honeycode : Automatisation InvokeScreen honeycode : Colonnes ListTable code d'origine : Rows ListTable code d'identification : ListTables code d'origine : Rows QueryTable honeycode : Job StartTable DataImport

Préfixe de service	Actions
iam	iam : AddClient ID ID ToOpen ConnectProvider iam : Profil AddRole ToInstance iam : AddUser ToGroup iam : Politique AttachGroup iam : Politique AttachRole iam : Politique AttachUser iam : ChangePassword iam : clé CreateAccess iam : Alias CreateAccount iam : CreateGroup iam : Profil CreateInstance iam : Profil CreateLogin iam : ID CreateOpen ConnectProvider iam : CreatePolicy iam : Version CreatePolicy iam : CreateRole iam:CreateSAMLProvider iam : CreateService LinkedRole iam : CreateService SpecificCredential iam : CreateUser iam : CreateVirtual MFAdevice

Préfixe de service	Actions
	iam:DeactivateMFADevice iam : clé DeleteAccess iam : Alias DeleteAccount iam : DeleteAccount PasswordPolicy iam : clé DeleteCloud FrontPublic iam : DeleteGroup iam : Politique DeleteGroup iam : Profil DeleteInstance iam : Profil DeleteLogin iam : ID DeleteOpen ConnectProvider iam : DeletePolicy iam : Version DeletePolicy iam : DeleteRole iam : DeleteRole PermissionsBoundary iam : Politique DeleteRole iam:DeleteSAMLProvider iam : Certificat DeleteServer iam : DeleteService LinkedRole iam : DeleteService SpecificCredential iam : Certificat DeleteSigning IAM : supprime SH PublicKey

Préfixe de service	Actions
	iam : DeleteUser iam : DeleteUser PermissionsBoundary iam : Politique DeleteUser iam : DeleteVirtual MFAdevice iam : Politique DetachGroup iam : Politique DetachRole iam : Politique DetachUser iam:EnableMFADevice iam : Rapport GenerateCredential iam : GenerateOrganizations AccessReport iam : Détails GenerateService LastAccessed iam : Usagé GetAccess KeyLast iam : GetAccount AuthorizationDetails iam : GetAccount EmailAddress iam : Nom GetAccount iam : GetAccount PasswordPolicy iam : Résumé GetAccount iam : clé GetCloud FrontPublic iam : GetContext KeysFor CustomPolicy iam : GetContext KeysFor PrincipalPolicy iam : Rapport GetCredential

Préfixe de service	Actions
	iam : GetGroup iam : Politique GetGroup iam : Profil GetInstance iam : Profil GetLogin iam:GetMFADevice iam : ID GetOpen ConnectProvider iam : GetOrganizations AccessReport iam : GetPolicy iam : Version GetPolicy iam : GetRole iam : Politique GetRole iam:GetSAMLProvider iam : Certificat GetServer iam : Détails GetService LastAccessed iam : Entités GetService LastAccessed DetailsWith iam : GetService LinkedRole DeletionStatus IAM : GetSSH PublicKey iam : GetUser iam : Politique GetUser iam : Clés ListAccess iam : Alias ListAccount

Préfixe de service	Actions
	iam : ListAttached GroupPolicies iam : ListAttached RolePolicies iam : ListAttached UserPolicies iam : Clés ListCloud FrontPublic iam : ListEntities ForPolicy iam : Politiques ListGroup iam : ListGroups iam : ListGroups ForUser iam : Profils ListInstance iam : Rôle ListInstance ProfilesFor iam:ListMFADevices iam : ID ListOpen ConnectProviders iam : ListPolicies iam : Accès ListPolicies GrantingService iam : Versions ListPolicy iam : Politiques ListRole iam : ListRoles iam:ListSAMLProviders iam : Certificats ListServer iam : ListService SpecificCredentials iam : Certificats ListSigning

Préfixe de service	Actions
	IAM : liste SSH PublicKeys IAM:ListSTS Status RegionalEndpoints iam : Politiques ListUser iam : ListUsers iam : ListVirtual MFADevices iam : Politique PutGroup iam : PutRole PermissionsBoundary iam : Politique PutRole iam : PutUser PermissionsBoundary iam : Politique PutUser iam : RemoveClient ID ID FromOpen ConnectProvider iam : Profil RemoveRole FromInstance iam : RemoveUser FromGroup iam : ResetService SpecificCredential iam:ResyncMFADevice iam : SetDefault PolicyVersion iam : Préférences SetSecurity TokenService IAM:définit le statut RegionalEndpoint de STS iam : Politique SimulateCustom iam : Politique SimulatePrincipal iam : clé UpdateAccess

Préfixe de service	Actions
	iam : UpdateAccount EmailAddress iam : Nom UpdateAccount iam : UpdateAccount PasswordPolicy iam : UpdateAssume RolePolicy iam : clé UpdateCloud FrontPublic iam : UpdateGroup iam : Profil UpdateLogin iam : UpdateOpen ID ConnectProvider Thumbprint iam : UpdateRole iam : Description UpdateRole iam:UpdateSAMLProvider iam : Certificat UpdateServer iam : UpdateService SpecificCredential iam : Certificat UpdateSigning IAM : met à jour SSH PublicKey iam : UpdateUser iam : clé UploadCloud FrontPublic iam : Certificat UploadServer iam : Certificat UploadSigning IAM : UploadSSH PublicKey

Préfixe de service	Actions
identitystore	boutique d'identité : CreateGroup identitystore : Adhésion CreateGroup boutique d'identité : CreateUser boutique d'identité : DeleteGroup identitystore : Adhésion DeleteGroup boutique d'identité : DeleteUser boutique d'identité : DescribeGroup identitystore : Adhésion DescribeGroup boutique d'identité : DescribeUser identitystore : Id GetGroup boutique d'identité : GetGroup MembershipId identitystore : Id GetUser boutique d'identité : IsMember InGroups identitystore : Abonnements ListGroup identitystore : Membre ListGroup MembershipsFor boutique d'identité : ListGroups boutique d'identité : ListUsers boutique d'identité : UpdateGroup boutique d'identité : UpdateUser

Préfixe de service	Actions
imagebuilder	imagebuilder : Création CancelImage imagebuilder : Exécution CancelLifecycle générateur d'images : CreateComponent imagebuilder : Recette CreateContainer imagebuilder : Configuration CreateDistribution générateur d'images : CreateImage imagebuilder : Pipeline CreateImage imagebuilder : Recette CreateImage imagebuilder : Configuration CreateInfrastructure imagebuilder : Politique CreateLifecycle générateur d'images : CreateWorkflow générateur d'images : DeleteComponent imagebuilder : Recette DeleteContainer imagebuilder : Configuration DeleteDistribution générateur d'images : DeleteImage imagebuilder : Pipeline DeleteImage imagebuilder : Recette DeleteImage imagebuilder : Configuration DeleteInfrastructure imagebuilder : Politique DeleteLifecycle générateur d'images : DeleteWorkflow imagebuilder : Politique GetComponent

Préfixe de service	Actions
	générateur d'images : GetContainer RecipePolicy imagebuilder : Politique GetImage générateur d'images : GetImage RecipePolicy imagebuilder : Exécution GetLifecycle imagebuilder : Politique GetLifecycle imagebuilder : Exécution GetWorkflow générateur d'images : GetWorkflow StepExecution générateur d'images : ImportComponent imagebuilder : ImportVm Image générateur d'images : ListComponent BuildVersions générateur d'images : ListComponents imagebuilder : Recettes ListContainer imagebuilder : Configurations ListDistribution générateur d'images : ListImage BuildVersions imagebuilder : Paquets ListImage générateur d'images : ListImage PipelineImages imagebuilder : Canalisations ListImage imagebuilder : Recettes ListImage générateur d'images : ListImages imagebuilder : Agrégations ListImage ScanFinding générateur d'images : ListImage ScanFindings

Préfixe de service	Actions
	imagebuilder : Configurations ListInfrastructure générateur d'images : ListLifecycle ExecutionResources imagebuilder : Exécutions ListLifecycle imagebuilder : Politiques ListLifecycle générateur d'images : ListWaiting WorkflowSteps imagebuilder : Exécutions ListWorkflow générateur d'images : ListWorkflows générateur d'images : ListWorkflow StepExecutions imagebuilder : Politique PutComponent générateur d'images : PutContainer RecipePolicy imagebuilder : Politique PutImage générateur d'images : PutImage RecipePolicy générateur d'images : SendWorkflow StepAction générateur d'images : StartImage PipelineExecution générateur d'images : StartResource StateUpdate imagebuilder : Configuration UpdateDistribution imagebuilder : Pipeline UpdateImage imagebuilder : Configuration UpdateInfrastructure

Préfixe de service	Actions
inspector	inspecteur : AddAttributes ToFindings
	inspecteur : CreateAssessment Target
	inspecteur : CreateAssessment Modèle
	inspecteur : CreateExclusions Aperçu
	inspecteur : CreateResource Groupe
	inspecteur : DeleteAssessment Exécuter
	inspecteur : DeleteAssessment Target
	inspecteur : DeleteAssessment Modèle
	inspecteur : DescribeAssessment Exécute
	inspecteur : DescribeAssessment Cibles
	inspector : DescribeAssessment Modèles
	inspecteur : DescribeCross AccountAccess Rôle
	inspecteur : DescribeExclusions
	inspecteur : DescribeFindings
	inspecteur : DescribeResource Groupes
	inspecteur : DescribeRules Packages
	inspecteur : GetAssessment Rapport
	inspecteur : GetExclusions Aperçu
	inspecteur : GetTelemetry Métadonnées
	inspecteur : ListAssessment RunAgents
	inspecteur : ListAssessment Exécute

Préfixe de service	Actions
	inspecteur : ListAssessment Cibles
	inspector : ListAssessment Modèles
	inspecteur : ListEvent Abonnements
	inspecteur : ListExclusions
	inspecteur : ListFindings
	inspecteur : ListRules Packages
	inspecteur : PreviewAgents
	inspecteur : RegisterCross AccountAccess Rôle
	inspecteur : RemoveAttributes FromFindings
	inspecteur : StartAssessment Exécuter
	inspecteur : StopAssessment Exécuter
	inspecteur : SubscribeTo Événement
	inspecteur : UnsubscribeFrom Événement
	inspecteur : UpdateAssessment Target

Préfixe de service	Actions
inspector2	inspecteur 2 : AssociateMember inspecteur 2 : BatchGet AccountStatus inspecteur 2 : BatchGet CodeSnippet inspecteur 2 : BatchGet FindingDetails inspector2 : Informations BatchGet FreeTrial inspector2 : 2 Statut BatchGet MemberEc DeepInspection inspector2 : 2 Statut BatchUpdate MemberEc DeepInspection inspector2 : Rapport CancelFindings inspector2 : Exportation CancelSbom inspecteur 2 : CreateCis ScanConfiguration inspecteur 2 : CreateFilter inspector2 : Rapport CreateFindings inspector2 : Exportation CreateSbom inspecteur 2 : DeleteCis ScanConfiguration inspecteur 2 : DeleteFilter inspector2 : Configuration DescribeOrganization inspector2:Disable inspecteur 2 : DisableDelegated AdminAccount inspecteur 2 : DisassociateMember inspector2:Enable inspecteur 2 : EnableDelegated AdminAccount

Préfixe de service	Actions
	inspecteur 2 : GetCis ScanReport inspector2 : Détails GetCis ScanResult inspecteur 2 : GetConfiguration inspecteur 2 : GetDelegated AdminAccount inspector2 : 2 Configuration GetEc DeepInspection inspector2 : Clé GetEncryption inspecteur 2 : GetFindings ReportStatus inspecteur 2 : GetMember inspector2 : Exportation GetSbom inspector2 : Autorisations ListAccount inspecteur 2 : ListCis ScanConfigurations inspector2 : Contrôles ListCis ScanResults AggregatedBy inspecteur 2 : ListCis ScanResults AggregatedBy TargetResource inspector2 : Scans ListCis inspecteur 2 : ListCoverage inspector2 : Statistiques ListCoverage inspecteur 2 : ListDelegated AdminAccounts inspecteur 2 : ListFilters inspector2 : Agrégations ListFinding inspecteur 2 : ListFindings inspecteur 2 : ListMembers

Préfixe de service	Actions
	inspector2 : Totaux ListUsage inspector2 : Clé ResetEncryption inspecteur 2 : SearchVulnerabilities inspecteur 2 : SendCis SessionHealth inspecteur 2 : SendCis SessionTelemetry inspector2 : Session StartCis inspector2 : Session StopCis inspecteur 2 : UpdateCis ScanConfiguration inspecteur 2 : UpdateConfiguration inspector2 : 2 Configuration UpdateEc DeepInspection inspector2 : Clé UpdateEncryption inspecteur 2 : UpdateFilter inspector2 : Configuration UpdateOrganization inspector2 : UpdateOrg Configuration d'Ec2 DeepInspection

Préfixe de service	Actions
iot	IoT : AcceptCertificate Transfert
	IoT : AddThing ToBilling Groupe
	IoT : AddThing ToThing Groupe
	IoT : AssociateTargets WithJob
	IoT : AttachPolicy
	IoT : AttachPrincipal Politique
	iot : AttachSecurity Profil
	IoT : AttachThing Principal
	IoT : CancelAudit MitigationActions Tâche
	IoT : CancelAudit Tâche
	IoT : CancelCertificate Transfert
	IoT : CancelDetect MitigationActions Tâche
	IoT : CancelJob
	IoT : CancelJob Exécution
	IoT : ClearDefault Autorisateur
	IoT : ConfirmTopic RuleDestination
	IoT : CreateAudit Suppression
	IoT : CreateAuthorizer
	IoT : CreateBilling Groupe
	IoT : CreateCertificate FromCsr
	iot : CreateCertificate Prestataire

Préfixe de service	Actions
	IoT : CreateCustom Metric
	IoT : CreateDimension
	IoT : CreateDomain Configuration
	IoT : CreateDynamic ThingGroup
	IoT : CreateFleet Metric
	IoT : CreateJob
	iot : CreateJob Modèle
	IoT : CreateKeys AndCertificate
	IoT : CreateMitigation Action
	iot:CreateOTAUpdate
	IoT : CreatePackage
	IoT : CreatePackage Version
	IoT : CreatePolicy
	IoT : CreatePolicy Version
	IoT : CreateProvisioning Réclamation
	iot : CreateProvisioning Modèle
	IoT : CreateProvisioning TemplateVersion
	Source : CreateRole Alias
	IoT : CreateScheduled Audit
	iot : CreateSecurity Profil
	IoT : CreateStream

Préfixe de service	Actions
	IoT : CreateThing
	IoT : CreateThing Groupe
	IoT : CreateThing Type
	IoT : CreateTopic Règle
	IoT : CreateTopic RuleDestination
	IoT : DeleteAccount AuditConfiguration
	IoT : DeleteAudit Suppression
	IoT : DeleteAuthorizer
	IoT : DeleteBilling Groupe
	iot:DeleteCACertificate
	IoT : DeleteCertificate
	iot : DeleteCertificate Prestataire
	IoT : DeleteCustom Metric
	IoT : DeleteDimension
	IoT : DeleteDomain Configuration
	IoT : DeleteDynamic ThingGroup
	IoT : DeleteFleet Metric
	IoT : DeleteJob
	IoT : DeleteJob Exécution
	iot : DeleteJob Modèle
	IoT : DeleteMitigation Action

Préfixe de service	Actions
	iot:DeleteOTAUpdate
	IoT : DeletePackage
	IoT : DeletePackage Version
	IoT : DeletePolicy
	IoT : DeletePolicy Version
	iot : DeleteProvisioning Modèle
	IoT : DeleteProvisioning TemplateVersion
	IoT : DeleteRegistration Code
	Source : DeleteRole Alias
	IoT : DeleteScheduled Audit
	iot : DeleteSecurity Profil
	IoT : DeleteStream
	IoT : DeleteThing
	IoT : DeleteThing Groupe
	IoT : DeleteThing Type
	IoT : DeleteTopic Règle
	IoT : DeleteTopic RuleDestination
	IoT : DeleteV2 LoggingLevel
	IoT : DeprecateThing Type
	IoT : DescribeAccount AuditConfiguration
	IoT : DescribeAudit Trouver

Préfixe de service	Actions
	IoT : DescribeAudit MitigationActions Tâche
	IoT : DescribeAudit Suppression
	IoT : DescribeAudit Tâche
	IoT : DescribeAuthorizer
	IoT : DescribeBilling Groupe
	iot:DescribeCACertificate
	IoT : DescribeCertificate
	iot : DescribeCertificate Prestataire
	IoT : DescribeCustom Metric
	IoT : DescribeDefault Autorisateur
	IoT : DescribeDetect MitigationActions Tâche
	IoT : DescribeDimension
	IoT : DescribeDomain Configuration
	IoT : DescribeEndpoint
	IoT : DescribeEvent Configurations
	IoT : DescribeFleet Metric
	IoT : DescribeIndex
	IoT : DescribeJob
	IoT : DescribeJob Exécution
	iot : DescribeJob Modèle
	IoT : DescribeManaged JobTemplate

Préfixe de service	Actions
	IoT : DescribeMitigation Action
	iot : DescribeProvisioning Modèle
	IoT : DescribeProvisioning TemplateVersion
	Source : DescribeRole Alias
	IoT : DescribeScheduled Audit
	iot : DescribeSecurity Profil
	IoT : DescribeStream
	IoT : DescribeThing
	IoT : DescribeThing Groupe
	IoT : DescribeThing RegistrationTask
	IoT : DescribeThing Type
	IoT : DetachPolicy
	IoT : DetachPrincipal Politique
	iot : DetachSecurity Profil
	IoT : DetachThing Principal
	IoT : DisableTopic Règle
	IoT : EnableTopic Règle
	iot : GetBehavior ModelTraining Résumés
	IoT : GetBuckets Agrégation
	IoT : GetCardinality
	IoT : GetEffective Politiques

Préfixe de service	Actions
	Objet : GetJob Document
	IoT : GetLogging Options
	iot:GetOTAUpdate
	IoT : GetPackage
	IoT : GetPackage Configuration
	IoT : GetPackage Version
	IoT : GetPercentiles
	IoT : GetPolicy
	IoT : GetPolicy Version
	IoT : GetRegistration Code
	IoT : GetStatistics
	IoT : GetTopic Règle
	IoT : GetTopic RuleDestination
	IoT : GetV2 LoggingOptions
	IoT : ListActive Violations
	IoT : ListAttached Politiques
	IoT : ListAudit résultats
	IoT : ListAudit MitigationActions Exécutions
	IoT : ListAudit MitigationActions Tâches
	IoT : ListAudit Suppressions
	IoT : ListAudit Tâches

Préfixe de service	Actions
	IoT : ListAuthorizers
	IoT : ListBilling Groupes
	iot:ListCACertificates
	IoT : ListCertificate Fournisseurs
	IoT : ListCertificates
	Lieu : ListCertificates ByCar
	IoT : ListCustom Métriques
	IoT : ListDetect MitigationActions Exécutions
	IoT : ListDetect MitigationActions Tâches
	IoT : ListDimensions
	IoT : ListDomain Configurations
	IoT : ListFleet Métriques
	IoT : ListIndices
	IoT : ListJob ExecutionsFor Job
	IoT : ListJob ExecutionsFor Chose
	IoT : ListJobs
	iot : ListJob Modèles
	IoT : ListManaged JobTemplates
	IoT : ListMetric Valeurs
	IoT : ListMitigation Actions
	iot:ListOTAUpdates

Préfixe de service	Actions
	iot : ListOutgoing Certificats
	IoT : ListPackages
	IoT : ListPackage Versions
	IoT : ListPolicies
	IoT : ListPolicy Principaux
	IoT : ListPolicy Versions
	IoT : ListPrincipal Politiques
	IoT : ListPrincipal Objets
	iot : ListProvisioning Modèles
	IoT : ListProvisioning TemplateVersions
	IoT : ListRelated ResourcesFor AuditFinding
	iot : ListRole Alias
	IoT : ListScheduled Audits
	iot : ListSecurity Profils
	IoT : ListSecurity ProfilesFor Cible
	IoT : ListStreams
	IoT : ListTargets ForPolicy
	iot : ListTargets ForSecurity Profil
	IoT : ListThing Groupes
	IoT : ListThing GroupsFor Chose
	IoT : ListThing Principaux

Préfixe de service	Actions
	IoT : ListThing RegistrationTask Rapports
	IoT : ListThing RegistrationTasks
	IoT : ListThings
	IoT : ListThings InBilling Groupe
	IoT : ListThings InThing Groupe
	IoT : ListThing Types
	IoT : ListTopic RuleDestinations
	IoT : ListTopic Règles
	IoT : Liste V2 LoggingLevels
	IoT : ListViolation Événements
	IoT : PutVerification StateOn Violation
	iot:RegisterCACertificate
	IoT : RegisterCertificate
	IoT : RegisterCertificate WithoutCam
	IoT : RegisterThing
	IoT : RejectCertificate Transfert
	IoT : RemoveThing FromBilling Groupe
	IoT : RemoveThing FromThing Groupe
	IoT : ReplaceTopic Règle
	IoT : SearchIndex
	IoT : SetDefault Autorisateur

Préfixe de service	Actions
	IoT : SetDefault PolicyVersion
	IoT : SetLogging Options
	IoT : SetV2 LoggingLevel
	IoT : SetV2 LoggingOptions
	IoT : StartAudit MitigationActions Tâche
	IoT : StartDetect MitigationActions Tâche
	IoT : StartOn DemandAudit Tâche
	IoT : StartThing RegistrationTask
	IoT : StopThing RegistrationTask
	IoT : TestAuthorization
	IoT : TestInvoke Autorisateur
	IoT : TransferCertificate
	IoT : UpdateAccount AuditConfiguration
	IoT : UpdateAudit Suppression
	IoT : UpdateAuthorizer
	IoT : UpdateBilling Groupe
	iot:UpdateCACertificate
	IoT : UpdateCertificate
	iot : UpdateCertificate Prestataire
	IoT : UpdateCustom Metric
	IoT : UpdateDimension

Préfixe de service	Actions
	IoT : UpdateDomain Configuration
	IoT : UpdateDynamic ThingGroup
	IoT : UpdateEvent Configurations
	IoT : UpdateFleet Metric
	IoT : UpdateIndexing Configuration
	IoT : UpdateJob
	IoT : UpdateMitigation Action
	IoT : UpdatePackage
	IoT : UpdatePackage Configuration
	IoT : UpdatePackage Version
	iot : UpdateProvisioning Modèle
	Source : UpdateRole Alias
	IoT : UpdateScheduled Audit
	iot : UpdateSecurity Profil
	IoT : UpdateStream
	IoT : UpdateThing
	IoT : UpdateThing Groupe
	IoT : UpdateThing GroupsFor Chose
	IoT : UpdateTopic RuleDestination
	IoT : ValidateSecurity ProfileBehaviors

Préfixe de service	Actions
iotanalytics	iotanalytics : Retraitement CancelPipeline Analyse de l'IoT : CreateChannel Analyse de l'IoT : CreateDataset iotanalytics : Contenu CreateDataset Analyse de l'IoT : CreateDatastore Analyse de l'IoT : CreatePipeline Analyse de l'IoT : DeleteChannel Analyse de l'IoT : DeleteDataset iotanalytics : Contenu DeleteDataset Analyse de l'IoT : DeleteDatastore Analyse de l'IoT : DeletePipeline Analyse de l'IoT : DescribeChannel Analyse de l'IoT : DescribeDataset Analyse de l'IoT : DescribeDatastore Bioanalyse : options DescribeLogging Analyse de l'IoT : DescribePipeline iotanalytics : Contenu GetDataset Analyse de l'IoT : ListChannels iotanalytics : Sommaire ListDataset Analyse de l'IoT : ListDatasets Analyse de l'IoT : ListDatastores

Préfixe de service	Actions
	Analyse de l'IoT : ListPipelines Bioanalyse : options PutLogging iotanalytics : Activité RunPipeline iotanalytics : données SampleChannel iotanalytics : Retraitement StartPipeline Analyse de l'IoT : UpdateChannel Analyse de l'IoT : UpdateDataset Analyse de l'IoT : UpdateDatastore Analyse de l'IoT : UpdatePipeline
iotdeviceadvisor	iotdeviceadvisor : Définition CreateSuite iotdeviceadvisor : Définition DeleteSuite conseiller pour appareils IoT : GetEndpoint iotdeviceadvisor : Définition GetSuite iotdeviceadvisor : Exécuter GetSuite conseiller pour appareils IoT : GetSuite RunReport iotdeviceadvisor : Définitions ListSuite iotdeviceadvisor : Exécute ListSuite iotdeviceadvisor : Exécuter StartSuite iotdeviceadvisor : Exécuter StopSuite iotdeviceadvisor : Définition UpdateSuite

Préfixe de service	Actions
iotevents	iotevents : Alarme BatchAcknowledge iotevents : Détecteur BatchDelete iotevents : Alarme BatchDisable iotevents : Alarme BatchEnable iotevents : Alarme BatchReset iotevents : Alarme BatchSnooze iotevents : Détecteur BatchUpdate iotevents : Modèle CreateAlarm iotevents : Modèle CreateDetector événements liés à l'Internet des objets : CreateInput iotevents : Modèle DeleteAlarm iotevents : Modèle DeleteDetector événements liés à l'Internet des objets : DeleteInput événements liés à l'Internet des objets : DescribeAlarm iotevents : Modèle DescribeAlarm événements liés à l'Internet des objets : DescribeDetector iotevents : Modèle DescribeDetector événements liés à l'Internet des objets : DescribeDetector ModelAnalysis événements liés à l'Internet des objets : DescribeInput iotevents : options DescribeLogging

Préfixe de service	Actions
	iotevents : Résultats GetDetector ModelAnalysis iotevents : Modèles ListAlarm événements liés à l'Internet des objets : ListAlarm ModelVersions événements liés à l'Internet des objets : ListAlarms iotevents : Modèles ListDetector événements liés à l'Internet des objets : ListDetector ModelVersions événements liés à l'Internet des objets : ListDetectors iotevents : Routages ListInput événements liés à l'Internet des objets : ListInputs iotevents : options PutLogging événements liés à l'Internet des objets : StartDetector ModelAnalysis iotevents : Modèle UpdateAlarm iotevents : Modèle UpdateDetector événements liés à l'Internet des objets : UpdateInput
iotfleethub	IoT Fleethub : CreateApplication IoT Fleethub : DeleteApplication IoT Fleethub : DescribeApplication IoT Fleethub : ListApplications IoT Fleethub : UpdateApplication

Préfixe de service	Actions
iotsitewise	à l'extérieur du site : AssociateAssets à l'extérieur du site : AssociateTime SeriesTo AssetProperty à l'extérieur du site : BatchAssociate ProjectAssets à l'extérieur du site : BatchDisassociate ProjectAssets à l'extérieur du site : valeur BatchGet AssetProperty à l'extérieur du site : BatchGet AssetProperty ValueHistory à l'extérieur du site : valeur BatchPut AssetProperty iotsitewise : Politique CreateAccess à l'extérieur du site : CreateAsset iotsitewise : Modèle CreateAsset iotsitewise : Modèle CreateAsset ModelComposite à l'extérieur du site : CreateBulk ImportJob à l'extérieur du site : CreateDashboard à l'extérieur du site : CreateGateway à l'extérieur du site : CreatePortal à l'extérieur du site : CreateProject iotsitewise : Politique DeleteAccess à l'extérieur du site : DeleteAsset iotsitewise : Modèle DeleteAsset iotsitewise : Modèle DeleteAsset ModelComposite à l'extérieur du site : DeleteDashboard

Préfixe de service	Actions
	à l'extérieur du site : DeleteGateway à l'extérieur du site : DeletePortal à l'extérieur du site : DeleteProject iotsitewise : Série DeleteTime iotsitewise : Politique DescribeAccess à l'extérieur du site : DescribeAsset à l'extérieur du site : DescribeAsset CompositeModel iotsitewise : Modèle DescribeAsset iotsitewise : Modèle DescribeAsset ModelComposite outsitewise : Propriété DescribeAsset à l'extérieur du site : DescribeBulk ImportJob à l'extérieur du site : DescribeDashboard à l'extérieur du site : DescribeDefault EncryptionConfiguration à l'extérieur du site : DescribeGateway à l'extérieur du site : DescribeGateway CapabilityConfiguration à l'extérieur du site : Options DescribeLogging à l'extérieur du site : DescribePortal à l'extérieur du site : DescribeProject à l'extérieur du site : Configuration DescribeStorage iotsitewise : Série DescribeTime à l'extérieur du site : DisassociateAssets

Préfixe de service	Actions
	à l'extérieur du site : DisassociateTime SeriesFrom AssetProperty à l'extérieur du site : ExecuteAction à l'extérieur du site : ExecuteQuery iotsitewise : Politiques ListAccess à l'extérieur du site : ListActions iotsitewise : Modèles ListAsset ModelComposite à l'extérieur du site : ListAsset ModelProperties iotsitewise : Modèles ListAsset iotsitewise : Propriétés ListAsset à l'extérieur du site : Relations ListAsset à l'extérieur du site : ListAssets à l'extérieur du site : actifs ListAssociated à l'extérieur du site : ListBulk ImportJobs à l'extérieur du site : Relations ListComposition à l'extérieur du site : ListDashboards à l'extérieur du site : ListGateways à l'extérieur du site : ListPortals à l'extérieur du site : actifs ListProject à l'extérieur du site : ListProjects iotsitewise : Série ListTime à l'extérieur du site : PutDefault EncryptionConfiguration

Préfixe de service	Actions
	à l'extérieur du site : Options PutLogging à l'extérieur du site : Configuration PutStorage iotsitewise : Politique UpdateAccess à l'extérieur du site : UpdateAsset iotsitewise : Modèle UpdateAsset iotsitewise : Modèle UpdateAsset ModelComposite outsitewise : Propriété UpdateAsset à l'extérieur du site : UpdateDashboard à l'extérieur du site : UpdateGateway à l'extérieur du site : UpdateGateway CapabilityConfiguration à l'extérieur du site : UpdatePortal à l'extérieur du site : UpdateProject

Préfixe de service	Actions
iottwinmaker	iottwinmaker : CancelMetadata TransferJob
	iottwinmaker : Type CreateComponent
	iottwinmaker : CreateEntity
	iottwinmaker : CreateMetadata TransferJob
	iottwinmaker : CreateScene
	iottwinmaker : Job CreateSync
	iottwinmaker : CreateWorkspace
	iottwinmaker : Type DeleteComponent
	iottwinmaker : DeleteEntity
	iottwinmaker : DeleteScene
	iottwinmaker : Job DeleteSync
	iottwinmaker : DeleteWorkspace
	iottwinmaker : ExecuteQuery
	iottwinmaker : GetMetadata TransferJob
	iottwinmaker : Plan GetPricing
	iottwinmaker : GetScene
	iottwinmaker : Job GetSync
	iottwinmaker : ListComponents
	iottwinmaker : Types ListComponent
	iottwinmaker : ListEntities
	iottwinmaker : ListMetadata TransferJobs

Préfixe de service	Actions
	iottwinmaker : ListProperties
	iottwinmaker : ListScenes
	iottwinmaker : Offres d'emploi ListSync
	iottwinmaker : Ressources ListSync
	iottwinmaker : ListWorkspaces
	iottwinmaker : Type UpdateComponent
	iottwinmaker : UpdateEntity
	iottwinmaker : Plan UpdatePricing
	iottwinmaker : UpdateScene
	iottwinmaker : UpdateWorkspace

Préfixe de service	Actions
iotwireless	IoT sans fil : AssociateAws AccountWith PartnerAccount IoT sans fil : AssociateMulticast GroupWith FuotaTask IoT sans fil : AssociateWireless DeviceWith FuotaTask IoT sans fil : AssociateWireless DeviceWith MulticastGroup IoT Wireless : Chose AssociateWireless DeviceWith iotwireless : Certificat AssociateWireless GatewayWith IoT Wireless : Chose AssociateWireless GatewayWith IoT sans fil : CancelMulticast GroupSession IoT sans fil : CreateDestination iotwireless : Profil CreateDevice iotwireless : Tâche CreateFuota IoT Wireless : Groupe CreateMulticast IoT sans fil : CreateNetwork AnalyzerConfiguration iotwireless : Profil CreateService iotwireless : Appareil CreateWireless IoT sans fil : Passerelle CreateWireless IoT sans fil : CreateWireless GatewayTask iotwireless : définition CreateWireless GatewayTask IoT sans fil : DeleteDestination iotwireless : Profil DeleteDevice iotwireless : Tâche DeleteFuota

Préfixe de service	Actions
	IoT Wireless : Groupe DeleteMulticast IoT sans fil : DeleteNetwork AnalyzerConfiguration iotwireless : Messages DeleteQueued iotwireless : Profil DeleteService iotwireless : Appareil DeleteWireless iotwireless : Tâche DeleteWireless DeviceImport IoT sans fil : Passerelle DeleteWireless IoT sans fil : DeleteWireless GatewayTask iotwireless : définition DeleteWireless GatewayTask iotwireless : Appareil DeregisterWireless IoT sans fil : DisassociateAws AccountFrom PartnerAccount IoT sans fil : DisassociateMulticast GroupFrom FuotaTask IoT sans fil : DisassociateWireless DeviceFrom FuotaTask IoT sans fil : DisassociateWireless DeviceFrom MulticastGroup IoT Wireless : Chose DisassociateWireless DeviceFrom iotwireless : Certificat DisassociateWireless GatewayFrom IoT Wireless : Chose DisassociateWireless GatewayFrom IoT sans fil : GetDestination iotwireless : Profil GetDevice IoT sans fil : GetEvent ConfigurationBy ResourceTypes iotwireless : Tâche GetFuota

Préfixe de service	Actions
	IoT sans fil : GetLog LevelsBy ResourceTypes iotwireless : Configuration GetMetric IoT sans fil : GetMetrics IoT Wireless : Groupe GetMulticast IoT sans fil : GetMulticast GroupSession IoT sans fil : GetNetwork AnalyzerConfiguration iotwireless : Compte GetPartner IoT sans fil : GetPosition iotwireless : Configuration GetPosition iotwireless : estimation GetPosition IoT sans fil : GetResource EventConfiguration IoT sans fil : GetResource LogLevel iotwireless : position GetResource iotwireless : point de terminaison GetService iotwireless : Profil GetService iotwireless : Appareil GetWireless iotwireless : Tâche GetWireless DeviceImport IoT sans fil : GetWireless DeviceStatistics IoT sans fil : Passerelle GetWireless IoT sans fil : GetWireless GatewayCertificate iotwireless : Informations GetWireless GatewayFirmware

Préfixe de service	Actions
	IoT sans fil : GetWireless GatewayStatistics IoT sans fil : GetWireless GatewayTask iotwireless : définition GetWireless GatewayTask IoT sans fil : ListDestinations iotwireless : Profils ListDevice iotwireless : Tâche ListDevices ForWireless DeviceImport IoT Wireless : Configurations ListEvent iotwireless : Tâches ListFuota iotwireless : Groupes ListMulticast IoT sans fil : ListMulticast GroupsBy FuotaTask IoT sans fil : ListNetwork AnalyzerConfigurations iotwireless : Comptes ListPartner IoT Wireless : Configurations ListPosition iotwireless : Messages ListQueued iotwireless : Profils ListService iotwireless : Tâches ListWireless DeviceImport iotwireless : Appareils ListWireless iotwireless : Passerelles ListWireless iotwireless : Définitions ListWireless GatewayTask iotwireless : Configuration PutPosition IoT sans fil : PutResource LogLevel

Préfixe de service	Actions
	iotwireless : 3 niveaux ResetAll ResourceLog IoT sans fil : ResetResource LogLevel IoT Wireless : Groupe SendData ToMulticast iotwireless : Appareil SendData ToWireless IoT sans fil : StartBulk AssociateWireless DeviceWith Multicast Group IoT sans fil : StartBulk DisassociateWireless DeviceFrom Multicast Group iotwireless : Tâche StartFuota IoT sans fil : StartMulticast GroupSession IoT sans fil : StartNetwork AnalyzerStream IoT sans fil : StartSingle WirelessDevice ImportTask iotwireless : Tâche StartWireless DeviceImport iotwireless : Appareil TestWireless IoT sans fil : UpdateDestination IoT sans fil : UpdateEvent ConfigurationBy ResourceTypes iotwireless : Tâche UpdateFuota IoT sans fil : UpdateLog LevelsBy ResourceTypes iotwireless : Configuration UpdateMetric IoT Wireless : Groupe UpdateMulticast IoT sans fil : UpdateNetwork AnalyzerConfiguration iotwireless : Compte UpdatePartner

Préfixe de service	Actions
	IoT sans fil : UpdatePosition IoT sans fil : UpdateResource EventConfiguration iotwireless : position UpdateResource iotwireless : Appareil UpdateWireless iotwireless : Tâche UpdateWireless DeviceImport IoT sans fil : Passerelle UpdateWireless

Préfixe de service	Actions
ivs	ivs : Canal BatchGet
	vis : BatchGet StreamKey
	ivs : Révocation BatchStart ViewerSession
	vis : CreateChannel
	ivs : Configuration CreateEncoder
	ivs : CreateParticipant Token
	vis : CreatePlayback RestrictionPolicy
	ivs : Configuration CreateRecording
	ivs : Configuration CreateStorage
	vis : Clé CreateStream
	vis : DeleteChannel
	ivs : Configuration DeleteEncoder
	vis : DeletePlayback KeyPair
	vis : DeletePlayback RestrictionPolicy
	ivs : Configuration DeleteRecording
	ivs : Configuration DeleteStorage
	vis : Clé DeleteStream
	vis : DisconnectParticipant
	vis : GetChannel
	vis : GetComposition
	ivs : Configuration GetEncoder

Préfixe de service	Actions
	vis : GetParticipant
	vis : GetPlayback KeyPair
	vis : GetPlayback RestrictionPolicy
	ivs : Configuration GetRecording
	ivs : Configuration GetStorage
	vis : GetStream
	vis : Clé GetStream
	ivs : Session GetStream
	vis : ImportPlayback KeyPair
	vis : ListChannels
	vis : ListCompositions
	ivs : Configurations ListEncoder
	ivs : Événements ListParticipant
	vis : ListParticipants
	vis : ListPlayback KeyPairs
	vis : ListPlayback RestrictionPolicies
	ivs : Configurations ListRecording
	ivs : Configurations ListStorage
	vis : Clés ListStream
	vis : ListStreams
	ivs : Sessions ListStream

Préfixe de service	Actions
	vis : PutMetadata vis : StartComposition vis : StartViewer SessionRevocation vis : StopComposition vis : StopStream vis : UpdateChannel vis : UpdatePlayback RestrictionPolicy
ivschat	ivschat : Token CreateChat type : Configuration CreateLogging visa : CreateRoom type : Configuration DeleteLogging visa : DeleteMessage visa : DeleteRoom visa : DisconnectUser type : Configuration GetLogging visa : GetRoom ivschat : Configurations ListLogging visa : ListRooms visa : SendEvent type : Configuration UpdateLogging visa : UpdateRoom

Préfixe de service	Actions
kafka	Kafka : BatchAssociate ScramSecret
	Kafka : BatchDisassociate ScramSecret
	Kafka : CreateCluster
	Source : V2 CreateCluster
	Kafka : CreateConfiguration
	Kafka : CreateReplicator
	Kafka : Connexion CreateVpc
	Kafka : DeleteCluster
	kafka : Politique DeleteCluster
	Kafka : DeleteConfiguration
	Kafka : DeleteReplicator
	Kafka : Connexion DeleteVpc
	Kafka : DescribeCluster
	Kafka : Opération DescribeCluster
	Kafka : OperationV2 DescribeCluster
	Source : V2 DescribeCluster
	Kafka : DescribeConfiguration
	kafka : Révision DescribeConfiguration
	Kafka : Connexion DescribeVpc
	Kafka : Brokers GetBootstrap
	kafka : Politique GetCluster

Préfixe de service	Actions
	Kafka : GetCompatible KafkaVersions
	Kafka : ListClient VpcConnections
	Kafka : Opérations ListCluster
	Kafka : Operations V2 ListCluster
	Kafka : ListClusters
	Source : V2 ListClusters
	kafka : Révisions ListConfiguration
	Kafka : ListConfigurations
	Kafka : Versions ListKafka
	Kafka : ListNodes
	Kafka : ListReplicators
	Kafka : Secrets ListScram
	Kafka : Connexions ListVpc
	kafka : Politique PutCluster
	Kafka : RebootBroker
	Kafka : RejectClient VpcConnection
	Kafka : comte UpdateBroker
	Kafka : Stockage UpdateBroker
	Kafka : Type UpdateBroker
	kafka : Configuration UpdateCluster
	Kafka : UpdateCluster KafkaVersion

Préfixe de service	Actions
	Kafka : UpdateConfiguration Kafka : UpdateConnectivity Kafka : UpdateMonitoring Kafka : Informations UpdateReplication Kafka : UpdateSecurity Kafka : UpdateStorage
kafkaconnect	Kafka Connect : CreateConnector kafkaconnect : Plug-in CreateCustom kafkaconnect : Configuration CreateWorker Kafka Connect : DeleteConnector kafkaconnect : Plug-in DeleteCustom kafkaconnect : Configuration DeleteWorker Kafka Connect : DescribeConnector kafkaconnect : Plug-in DescribeCustom kafkaconnect : Configuration DescribeWorker Kafka Connect : ListConnectors kafkaconnect : Plug-ins ListCustom kafkaconnect : Configurations ListWorker Kafka Connect : UpdateConnector

Préfixe de service	Actions
kendra	kendra : AssociateEntities ToExperience
	kendra : AssociatePersonas ToEntities
	Kendra : Document BatchDelete
	Kendra : Set BatchDelete FeaturedResults
	kendra : BatchGet DocumentStatus
	Kendra : Document BatchPut
	kendra : Suggestions ClearQuery
	kendra : CreateAccess ControlConfiguration
	calendrier : Source CreateData
	kendra : CreateExperience
	kendra : CreateFaq
	kendra : CreateFeatured ResultsSet
	kendra : CreateIndex
	kendra : Liste CreateQuery SuggestionsBlock
	kendra : CreateThesaurus
	Kendra : Source DeleteData
	kendra : DeleteExperience
	kendra : DeleteFaq
	kendra : DeleteIndex
	kendra : Cartographie DeletePrincipal
	kendra : Liste DeleteQuery SuggestionsBlock

Préfixe de service	Actions
	kendra : DeleteThesaurus
	kendra : DescribeAccess ControlConfiguration
	Kendra : Source DescribeData
	kendra : DescribeExperience
	kendra : DescribeFaq
	kendra : DescribeFeatured ResultsSet
	kendra : DescribeIndex
	kendra : Cartographie DescribePrincipal
	kendra : Liste DescribeQuery SuggestionsBlock
	kendra : DescribeQuery SuggestionsConfig
	kendra : DescribeThesaurus
	kendra : DisassociateEntities FromExperience
	kendra : DisassociatePersonas FromEntities
	kendra : Suggestions GetQuery
	kendra : GetSnapshots
	kendra : ListAccess ControlConfigurations
	kendra : Sources ListData
	kendra : ListData SourceSync Offres d'emploi
	kendra : Personas ListEntity
	kendra : Entities ListExperience
	kendra : ListExperiences

Préfixe de service	Actions
	kendra : ListFaqs
	kendra : ListFeatured ResultsSets
	kendra : ListGroups OlderThan OrderingId
	kendra : ListIndices
	kendra : Listes ListQuery SuggestionsBlock
	kendra : ListThesauri
	kendra : Cartographie PutPrincipal
	kendra:Query
	kendra:Retrieve
	kendra : Job StartData SourceSync
	kendra : Job StopData SourceSync
	kendra : SubmitFeedback
	Kendra : Source UpdateData
	kendra : UpdateExperience
	kendra : UpdateFeatured ResultsSet
	kendra : UpdateIndex
	kendra : Liste UpdateQuery SuggestionsBlock
	kendra : UpdateQuery SuggestionsConfig
	kendra : UpdateThesaurus

Préfixe de service	Actions
kinesis	kinésie : CreateStream kinésie : DecreaseStream RetentionPeriod kinésie : DeleteStream kinesis : Consommateur DeregisterStream kinésie : DescribeLimits kinésie : DescribeStream kinesis : Consommateur DescribeStream kinesis : Résumé DescribeStream kinesis : Surveillance DisableEnhanced kinesis : Surveillance EnableEnhanced kinésie : IncreaseStream RetentionPeriod kinésie : ListShards kinesis : Consommateurs ListStream kinésie : ListStreams kinésie : MergeShards kinesis : Consommateur RegisterStream kinésie : SplitShard kinesis : Chiffrement StartStream kinesis : Chiffrement StopStream kinesis : Count UpdateShard kinésie : Mode UpdateStream

Préfixe de service	Actions
kinesisanalytics	Kinesis Analytics : AddApplication CloudWatch LoggingOption
	kinesisanalytics : Entrée AddApplication
	kinesisanalytics : Configuration AddApplication InputProcessing
	kinesisanalytics : sortie AddApplication
	kinesisanalytics : Source AddApplication ReferenceData
	Kinesis Analytics : AddApplication VpcConfiguration
	Kinesis Analytics : CreateApplication
	Kinesis Analytics : CreateApplication PresignedUrl
	kinesisanalytics : Instantané CreateApplication
	Kinesis Analytics : DeleteApplication
	Kinesis Analytics : DeleteApplication CloudWatch LoggingOption
	kinesisanalytics : Configuration DeleteApplication InputProcessing
	kinesisanalytics : sortie DeleteApplication
	kinesisanalytics : Source DeleteApplication ReferenceData
	kinesisanalytics : Instantané DeleteApplication
	Kinesis Analytics : DeleteApplication VpcConfiguration
	Kinesis Analytics : DescribeApplication
	kinesisanalytics : Instantané DescribeApplication
	kinesisanalytics : Version DescribeApplication
	kinesisanalytics : Schéma DiscoverInput
	Kinesis Analytics : ListApplications

Préfixe de service	Actions
	kinesisanalytics : Instantanés ListApplication
	kinesisanalytics : Versions ListApplication
	Kinesis Analytics : RollbackApplication
	Kinesis Analytics : StartApplication
	Kinesis Analytics : StopApplication
	Kinesis Analytics : UpdateApplication
	Kinesis Analytics : UpdateApplication MaintenanceConfiguration

Préfixe de service	Actions
kms	kms : CancelKey Suppression km : ConnectCustom KeyStore km : CreateAlias km : CreateCustom KeyStore km : CreateGrant km : CreateKey kms:Decrypt km : DeleteAlias km : DeleteCustom KeyStore km : DeleteImported KeyMaterial km : DescribeCustom KeyStores km : DescribeKey km : DisableKey km : DisableKey Rotation km : DisconnectCustom KeyStore km : EnableKey km : EnableKey Rotation kms:Encrypt km : GenerateData Clé km : GenerateData KeyPair km : GenerateData KeyPair WithoutPlaintext

Préfixe de service	Actions
	kms : Texte GenerateData KeyWithout en clair km : GenerateMac km : GenerateRandom kms : GetKey Politique km : GetKey RotationStatus km : GetParameters ForImport km : GetPublic Clé km : ImportKey Matériau km : ListAliases km : ListGrants kms : ListKey Politiques km : ListKeys kms : ListRetirable Subventions km : ReplicateKey km : RetireGrant km : RevokeGrant kms : ScheduleKey Suppression kms:Sign km : UpdateAlias km : UpdateCustom KeyStore km : UpdateKey Description

Préfixe de service	Actions
	km : UpdatePrimary Région kms:Verify km : VerifyMac

Préfixe de service	Actions
lambda	lambda : AddLayer VersionPermission
	lambda : AddLayer VersionPermission
	lambda : AddPermission
	lambda : AddPermission
	lambda : AddPermission
	lambda : CreateAlias
	lambda : CreateAlias
	lambda : CreateCode SigningConfig
	lambda : CreateEvent SourceMapping
	lambda : CreateEvent SourceMapping
	lambda : CreateFunction
	lambda : CreateFunction
	lambda : CreateFunction UrlConfig
	lambda : DeleteAlias
	lambda : DeleteAlias
	lambda : DeleteCode SigningConfig
	lambda : DeleteEvent SourceMapping
	lambda : DeleteEvent SourceMapping
	lambda : DeleteFunction
	lambda : DeleteFunction
	lambda : Config DeleteFunction CodeSigning

Préfixe de service	Actions
	lambda : Concurrence DeleteFunction
	lambda : Concurrence DeleteFunction
	lambda : Config DeleteFunction EventInvoke
	lambda : DeleteFunction UrlConfig
	lambda : Version DeleteLayer
	lambda : Version DeleteLayer
	lambda : DeleteProvisioned ConcurrencyConfig
	lambda : Paramètres GetAccount
	lambda : Paramètres GetAccount
	lambda : GetAlias
	lambda : GetAlias
	lambda : GetCode SigningConfig
	lambda : GetEvent SourceMapping
	lambda : GetEvent SourceMapping
	lambda : GetFunction
	lambda : GetFunction
	lambda : GetFunction
	lambda : Config GetFunction CodeSigning
	lambda : Concurrence GetFunction
	lambda : Configuration GetFunction
	lambda : Configuration GetFunction

Préfixe de service	Actions
	lambda : Configuration GetFunction
	lambda : Config GetFunction EventInvoke
	lambda : GetFunction UrlConfig
	lambda : Version GetLayer
	lambda : Version GetLayer
	lambda : Version GetLayer
	lambda : Version GetLayer
	lambda : GetLayer VersionPolicy
	lambda : GetLayer VersionPolicy
	lambda : GetPolicy
	lambda : GetPolicy
	lambda : GetPolicy
	lambda : GetProvisioned ConcurrencyConfig
	lambda : GetRuntime ManagementConfig
	lambda : ListAliases
	lambda : ListAliases
	lambda : ListCode SigningConfigs
	lambda : ListEvent SourceMappings
	lambda : ListEvent SourceMappings
	lambda : Configurations ListFunction EventInvoke
	lambda : ListFunctions

Préfixe de service	Actions
	lambda : ListFunctions
	lambda : ListFunctions ByCode SigningConfig
	lambda : ListFunction UrlConfigs
	lambda : ListLayers
	lambda : ListLayers
	lambda : Versions ListLayer
	lambda : Versions ListLayer
	lambda : ListProvisioned ConcurrencyConfigs
	lambda : ListVersions ByFunction
	lambda : ListVersions ByFunction
	lambda : Version PublishLayer
	lambda : Version PublishLayer
	lambda : PublishVersion
	lambda : PublishVersion
	lambda : Config PutFunction CodeSigning
	lambda : Concurrence PutFunction
	lambda : Concurrence PutFunction
	lambda : Config PutFunction EventInvoke
	lambda : PutProvisioned ConcurrencyConfig
	lambda : PutRuntime ManagementConfig
	lambda : RemoveLayer VersionPermission

Préfixe de service	Actions
	lambda : RemoveLayer VersionPermission
	lambda : RemovePermission
	lambda : RemovePermission
	lambda : RemovePermission
	lambda : UpdateAlias
	lambda : UpdateAlias
	lambda : UpdateCode SigningConfig
	lambda : UpdateEvent SourceMapping
	lambda : UpdateEvent SourceMapping
	lambda : Code UpdateFunction
	lambda : Code UpdateFunction
	lambda : Code UpdateFunction
	lambda : Configuration UpdateFunction
	lambda : Configuration UpdateFunction
	lambda : Configuration UpdateFunction
	lambda : Config UpdateFunction EventInvoke
	lambda : UpdateFunction UrlConfig

Préfixe de service	Actions
lex	Lex : BatchCreate CustomVocabulary Objet Lex : BatchDelete CustomVocabulary Objet Lex : BatchUpdate CustomVocabulary Objet lex : BuildBot Local Alex : CreateBot Alias Lex : CreateBot Version Alex : CreateExport Lex : CreateIntent Version lex : CreateResource Politique Alex : CreateSlot TypeVersion lex : CreateTest SetDiscrepancy Rapport lex : CreateUpload URL Alex : DeleteBot Alex : DeleteBot ChannelAssociation Alex : DeleteExport Alex : DeleteImport Lex : DeleteIntent Version lex : DeleteResource Politique Alex : DeleteSlot TypeVersion Alex : DeleteTest Set Alex : DeleteUtterances

Préfixe de service	Actions
	Alex : DescribeBot Alias lex : DescribeBot Recommandation Alex : DescribeBot ResourceGeneration Lex : DescribeBot Version Alex : DescribeCustom VocabularyMetadata Alex : DescribeExport Alex : DescribeImport lex : DescribeResource Politique lex : DescribeTest Exécution Alex : DescribeTest Set lex : DescribeTest SetDiscrepancy Rapport Alex : DescribeTest SetGeneration lex : GenerateBot Elément Alex : GetBot Alex : GetBot Alias lex : GetBot Alias Alex : GetBot ChannelAssociation Alex : GetBot ChannelAssociations Alex : GetBots Lex : GetBot Versions lex : GetBuiltin Intention

Préfixe de service	Actions
	Alex : GetBuiltin Intentions Alex : GetBuiltin SlotTypes Alex : GetExport Alex : GetImport Alex : GetIntent Alex : GetIntents Lex : GetIntent Versions Alex : GetMigration Alex : GetMigrations Lex : GetSlot Type Lex : GetSlot Types Alex : GetSlot TypeVersions lex : GetTest ExecutionArtifacts URL lex : GetUtterances Afficher lex : ListBot Alias lex : ListBot Recommandations Alex : ListBot ResourceGenerations Alex : ListBots Lex : ListBot Versions Alex : ListBuilt InIntents Lex : ListBuilt InSlot Types

Préfixe de service	Actions
	Alex : ListCustom VocabularyItems Alex : ListExports Alex : ListImports Lex : ListIntent Métriques Lex : ListIntent Chemins Alex : ListRecommended Intentions Alex : ListSession AnalyticsData Lex : ListSession Métriques Lex : 2 ListTest ExecutionResult objets Lex : ListTest Exécutions lex : ListTest Ensembles Alex : PutBot Alex : PutBot Alias Alex : PutIntent Lex : PutSlot Type lex : SearchAssociated Transcriptions lex : StartBot Recommandation Alex : StartImport Alex : StartMigration lex : StartTest Exécution Alex : StartTest SetGeneration

Préfixe de service	Actions
	lex : StopBot Recommandation Alex : UpdateBot Alias lex : UpdateBot Recommandation Alex : UpdateExport lex : UpdateResource Politique
license-manager-linux-subscriptions	license-manager-linux-subscription : Paramètres GetService License-Manager-Linux-Subscriptions : ListLinux SubscriptionInstances license-manager-linux-subscription : Abonnements ListLinux license-manager-linux-subscription : Paramètres UpdateService

Préfixe de service	Actions
lightsail	voile optique : IP AllocateStatic voile lumineuse : AttachCertificate ToDistribution voile lumineuse : AttachDisk lightsail : Équilibreur AttachInstances ToLoad lightsail : Certificat AttachLoad BalancerTls voile optique : IP AttachStatic voile lumineuse : CloseInstance PublicPorts voile lumineuse : CopySnapshot voile lumineuse : CreateBucket voile lumineuse : CreateBucket AccessKey voile lumineuse : CreateCertificate voile lumineuse : CreateCloud FormationStack lightsail : Méthode CreateContact lightsail : Service CreateContainer voile lumineuse : CreateContainer ServiceDeployment lightsail : Connexion CreateContainer ServiceRegistry voile lumineuse : CreateDisk voile lumineuse : CreateDisk FromSnapshot lightsail : Instantané CreateDisk voile lumineuse : CreateDistribution voile lumineuse : CreateDomain

Préfixe de service	Actions
	LightSail SessionAccess : CreateGUI Details voile lumineuse : CreateInstances voile lumineuse : CreateInstances FromSnapshot lightsail : Instantané CreateInstance lightsail : paire CreateKey lightsail : Équilibreur CreateLoad lightsail : Certificat CreateLoad BalancerTls lightsail : Base de données CreateRelational lightsail : Instantané CreateRelational DatabaseFrom voile lumineuse : CreateRelational DatabaseSnapshot voile lumineuse : DeleteAlarm lightsail : Instantané DeleteAuto voile lumineuse : DeleteBucket voile lumineuse : DeleteBucket AccessKey voile lumineuse : DeleteCertificate lightsail : Méthode DeleteContact voile lumineuse : Image DeleteContainer lightsail : Service DeleteContainer voile lumineuse : DeleteDisk lightsail : Instantané DeleteDisk voile lumineuse : DeleteDistribution

Préfixe de service	Actions
	voile lumineuse : DeleteDomain lightsail : Entrée DeleteDomain voile lumineuse : DeleteInstance lightsail : Instantané DeleteInstance lightsail : paire DeleteKey voile lumineuse : DeleteKnown HostKeys lightsail : Équilibreur DeleteLoad lightsail : Certificat DeleteLoad BalancerTls lightsail : Base de données DeleteRelational voile lumineuse : DeleteRelational DatabaseSnapshot voile lumineuse : DetachCertificate FromDistribution voile lumineuse : DetachDisk lightsail : Équilibreur DetachInstances FromLoad voile optique : IP DetachStatic LightSail : Activé DisableAdd voile lumineuse : DownloadDefault KeyPair LightSail : Activé EnableAdd voile lumineuse : ExportSnapshot lightsail : Noms GetActive voile lumineuse : GetAlarms lightsail : Instantanés GetAuto

Préfixe de service	Actions
	voile lumineuse : GetBlueprints voile lumineuse : GetBucket AccessKeys lightsail : Bundles GetBucket voile lumineuse : GetBucket MetricData voile lumineuse : GetBuckets voile lumineuse : GetBundles voile lumineuse : GetCertificates lightsail : records GetCloud FormationStack lightsail : Méthodes GetContact lightsail : métadonnées d'API GetContainer voile lumineuse : Images GetContainer lightsail : journal GetContainer voile lumineuse : GetContainer ServiceDeployments lightsail : Données GetContainer ServiceMetric voile lumineuse : GetContainer ServicePowers lightsail : Services GetContainer lightsail : estimation GetCost voile lumineuse : GetDisk voile lumineuse : GetDisks lightsail : Instantané GetDisk lightsail : Instantanés GetDisk

Préfixe de service	Actions
	lightsail : Bundles GetDistribution lightsail : Réinitialiser GetDistribution LatestCache voile lumineuse : GetDistribution MetricData voile lumineuse : GetDistributions voile lumineuse : GetDomain voile lumineuse : GetExport SnapshotRecords voile lumineuse : GetInstance voile lumineuse : GetInstance MetricData voile lumineuse : GetInstance PortStates voile lumineuse : GetInstances lightsail : Instantané GetInstance lightsail : Instantanés GetInstance lightsail : État GetInstance lightsail : paire GetKey lightsail : Paris GetKey lightsail : Équilibreur GetLoad lightsail : Données GetLoad BalancerMetric lightsail : Équilibreurs GetLoad lightsail : Certificats GetLoad BalancerTls lightsail : Politiques GetLoad BalancerTls voile lumineuse : GetOperation

Préfixe de service	Actions
	voile lumineuse : GetOperations voile lumineuse : GetOperations ForResource voile lumineuse : GetRegions lightsail : Base de données GetRelational voile lumineuse : GetRelational DatabaseBlueprints voile lumineuse : GetRelational DatabaseBundles voile lumineuse : GetRelational DatabaseEvents lightsail : Événements GetRelational DatabaseLog lightsail : Streams GetRelational DatabaseLog voile lumineuse : GetRelational DatabaseMaster UserPassword lightsail : Données GetRelational DatabaseMetric voile lumineuse : GetRelational DatabaseParameters lightsail : Bases de données GetRelational voile lumineuse : GetRelational DatabaseSnapshot voile lumineuse : GetRelational DatabaseSnapshots lightsail : Histoire GetSetup voile optique : IP GetStatic voile lumineuse : Ips GetStatic lightsail : paire ImportKey voile lumineuse : Peered IsVpc voile lumineuse : OpenInstance PublicPorts

Préfixe de service	Actions
	voile lumineuse : PeerVpc voile lumineuse : PutAlarm voile lumineuse : PutInstance PublicPorts voile lumineuse : RebootInstance lightsail : Base de données RebootRelational voile lumineuse : Image RegisterContainer voile optique : IP ReleaseStatic lightsail : cache ResetDistribution voile lumineuse : SendContact MethodVerification voile lumineuse : SetIp AddressType lightsail : seau SetResource AccessFor LightSail : Https SetupInstance lightsail:StartGUISession voile lumineuse : StartInstance lightsail : Base de données StartRelational lightsail:StopGUISession voile lumineuse : StopInstance lightsail : Base de données StopRelational voile lumineuse : TestAlarm voile lumineuse : UnpeerVpc voile lumineuse : UpdateBucket

Préfixe de service	Actions
	lightsail : Bundle UpdateBucket lightsail : Service UpdateContainer voile lumineuse : UpdateDistribution lightsail : Bundle UpdateDistribution lightsail : Entrée UpdateDomain voile lumineuse : UpdateInstance MetadataOptions voile lumineuse : UpdateLoad BalancerAttribute lightsail : Base de données UpdateRelational voile lumineuse : UpdateRelational DatabaseParameters

Préfixe de service	Actions
journaux	journaux : AssociateKms clé
	logs : CancelExport Tâche
	logs : CreateExport Tâche
	journaux : CreateLog AnomalyDetector
	logs : CreateLog Groupe
	journaux : CreateLog Stream
	journaux : DeleteData ProtectionPolicy
	journaux : DeleteDelivery
	journaux : DeleteDelivery Destination
	journaux : DeleteDelivery DestinationPolicy
	journaux : DeleteDelivery Source
	journaux : DeleteDestination
	logs : DeleteLog Groupe
	journaux : DeleteLog Stream
	logs : DeleteMetric Filtre
	logs : DeleteQuery définition
	logs : DeleteResource Politique
	logs : DeleteRetention Politique
	logs : DeleteSubscription Filtre
	journaux : DescribeAccount Politiques
	journaux : DescribeDeliveries

Préfixe de service	Actions
	journaux : DescribeDelivery Destinations
	journaux : DescribeDelivery Sources
	journaux : DescribeDestinations
	journaux : DescribeExport Tâches
	journaux : DescribeLog Groupes
	journaux : DescribeLog Streams
	journaux : DescribeMetric Filtres
	journaux : DescribeQueries
	logs : DescribeQuery Définitions
	journaux : DescribeResource Politiques
	journaux : DescribeSubscription Filtres
	journaux : DisassociateKms clé
	journaux : GetData ProtectionPolicy
	journaux : GetDelivery
	journaux : GetDelivery Destination
	journaux : GetDelivery DestinationPolicy
	journaux : GetDelivery Source
	journaux : GetLog GroupFields
	journaux : GetLog Enregistrer
	logs : GetQuery Résultats
	journaux : ListAnomalies

Préfixe de service	Actions
	journaux : ListLog AnomalyDetectors
	journaux : PutData ProtectionPolicy
	journaux : PutDelivery Destination
	journaux : PutDelivery DestinationPolicy
	journaux : PutDelivery Source
	journaux : PutDestination
	logs : PutDestination Politique
	logs : PutMetric Filtre
	logs : PutQuery définition
	logs : PutResource Politique
	logs : PutRetention Politique
	logs : PutSubscription Filtre
	Blogs : StartLive Tail
	journaux : StartQuery
	journaux : StopQuery
	logs : TestMetric Filtre

Préfixe de service	Actions
lookoutequipment	équipement de surveillance : CreateDataset équipement de surveillance : Scheduler CreateInference équipement de surveillance : CreateLabel équipement de surveillance : Groupe CreateLabel équipement de surveillance : CreateModel équipement de surveillance : DeleteDataset équipement de surveillance : Scheduler DeleteInference équipement de surveillance : DeleteLabel équipement de surveillance : Groupe DeleteLabel équipement de surveillance : DeleteModel lookouteequipment : Politique DeleteResource équipement de surveillance : Scheduler DeleteRetraining équipement de surveillance : DescribeData IngestionJob équipement de surveillance : DescribeDataset équipement de surveillance : Scheduler DescribeInference lookoutequipment:Describelabel équipement de surveillance : Groupe DescribeLabel équipement de surveillance : DescribeModel équipement de surveillance : Version DescribeModel lookouteequipment : Politique DescribeResource équipement de surveillance : Scheduler DescribeRetraining

Préfixe de service	Actions
	équipement de surveillance : ImportDataset équipement de surveillance : Version ImportModel équipement de surveillance : ListData IngestionJobs équipement de surveillance : ListDatasets lookoutequipment : Événements ListInference équipement de surveillance : Exécutions ListInference équipement de surveillance : Planificateurs ListInference équipement de surveillance : Groupes ListLabel équipement de surveillance : ListLabels équipement de surveillance : ListModels équipement de surveillance : Versions ListModel équipement de surveillance : Planificateurs ListRetraining lookoutequipment : Statistiques ListSensor lookouteequipment : Politique PutResource équipement de surveillance : StartData IngestionJob équipement de surveillance : Scheduler StartInference équipement de surveillance : Scheduler StartRetraining équipement de surveillance : Scheduler StopInference équipement de surveillance : Scheduler StopRetraining équipement de surveillance : UpdateActive ModelVersion équipement de surveillance : Scheduler UpdateInference

Préfixe de service	Actions
	équipement de surveillance : Groupe UpdateLabel équipement de surveillance : UpdateModel équipement de surveillance : Scheduler UpdateRetraining

Préfixe de service	Actions
lookoutmetrics	lookoutmetrics : Détecteur ActivateAnomaly
	Métriques de surveillance : BackTest AnomalyDetector
	Métriques de surveillance : CreateAlert
	lookoutmetrics : Détecteur CreateAnomaly
	lookoutmetrics : Set CreateMetric
	lookoutmetrics : Détecteur DeactivateAnomaly
	Métriques de surveillance : DeleteAlert
	lookoutmetrics : Détecteur DeleteAnomaly
	Métriques de surveillance : DescribeAlert
	Métriques de surveillance : DescribeAnomaly DetectionExecutions
	lookoutmetrics : Détecteur DescribeAnomaly
	lookoutmetrics : Set DescribeMetric
	Métriques de surveillance : DetectMetric SetConfig
	lookoutmetrics : Groupe GetAnomaly
	Métriques de surveillance : GetData QualityMetrics
	Métriques de surveillance : GetFeedback
	lookoutmetrics : Données GetSample
	Métriques de surveillance : ListAlerts
	lookoutmetrics : Détecteurs ListAnomaly
	lookoutmetrics : Métriques ListAnomaly GroupRelated
	Métriques de surveillance : ListAnomaly GroupSummaries

Préfixe de service	Actions
	lookoutmetrics : Série ListAnomaly GroupTime
	lookoutmetrics : Ensembles ListMetric
	Métriques de surveillance : PutFeedback
	Métriques de surveillance : UpdateAlert
	lookoutmetrics : Détecteur UpdateAnomaly
	lookoutmetrics : Set UpdateMetric

Préfixe de service	Actions
lookoutvision	vision de l'œil : CreateDataset vision de l'œil : CreateModel vision de l'œil : CreateProject vision de l'œil : DeleteDataset vision de l'œil : DeleteModel vision de l'œil : DeleteProject vision de l'œil : DescribeDataset vision de l'œil : DescribeModel vision de l'œil : DescribeModel PackagingJob vision de l'œil : DescribeProject vision de l'œil : DetectAnomalies lookoutvision : Entrées ListDataset vision de l'œil : ListModel PackagingJobs vision de l'œil : ListModels vision de l'œil : ListProjects vision de l'œil : StartModel vision de l'œil : StartModel PackagingJob vision de l'œil : StopModel lookoutvision : Entrées UpdateDataset

Préfixe de service	Actions
m2	m2 : CancelBatch JobExecution
	m2 : CreateApplication
	m2 : CreateData SetImport Tâche
	m2 : CreateDeployment
	m2 : CreateEnvironment
	m2 : DeleteApplication
	m2 : DeleteApplication FromEnvironment
	m2 : DeleteEnvironment
	m2 : GetApplication
	m2 : GetApplication Version
	m2 : GetBatch JobExecution
	m2 : GetData SetDetails
	m2 : GetData SetImport Tâche
	m2 : GetDeployment
	m2 : GetEnvironment
	m2 : GetSigned BluinsightsUrl
	m2 : ListApplications
	m2 : ListApplication Versions
	m2 : ListBatch JobDefinitions
	m2 : ListBatch JobExecutions
	m2 : ListBatch JobRestart Points

Préfixe de service	Actions
	m2 : ListData SetImport Histoire
	m2 : ListData Ensembles
	m2 : ListDeployments
	m2 : ListEngine Versions
	m2 : ListEnvironments
	m2 : StartApplication
	m2 : StartBatch Job
	m2 : StopApplication
	m2 : UpdateApplication
	m2 : UpdateEnvironment

Préfixe de service	Actions
managedblockchain	blockchain gérée : CreateAccessor blockchain gérée : CreateMember blockchain gérée : CreateNetwork blockchain gérée : CreateNode blockchain gérée : CreateProposal blockchain gérée : DeleteAccessor blockchain gérée : DeleteMember blockchain gérée : DeleteNode blockchain gérée : GetAccessor blockchain gérée : GetMember blockchain gérée : GetNetwork blockchain gérée : GetNode blockchain gérée : GetProposal blockchain gérée : InvokeRpc PolygonMainnet blockchain gérée : Testnet InvokeRpc PolygonMumbai blockchain gérée : ListAccessors blockchain gérée : ListInvitations blockchain gérée : ListMembers blockchain gérée : ListNetworks blockchain gérée : ListNodes blockchain gérée : ListProposals

Préfixe de service	Actions
	blockchain gérée : Votes ListProposal blockchain gérée : RejectInvitation blockchain gérée : UpdateMember blockchain gérée : UpdateNode blockchain gérée : Proposition VoteOn

Préfixe de service	Actions
mediaconnect	mediaconnect : Sorties AddBridge
	mediaconnect : Sources AddBridge
	connexion multimédia : AddFlow MediaStreams
	mediaconnect : Sorties AddFlow
	mediaconnect : Sources AddFlow
	connexion multimédia : AddFlow VpcInterfaces
	connexion multimédia : CreateBridge
	connexion multimédia : CreateFlow
	connexion multimédia : CreateGateway
	connexion multimédia : DeleteBridge
	connexion multimédia : DeleteFlow
	connexion multimédia : DeleteGateway
	mediaconnect : Instance DeregisterGateway
	connexion multimédia : DescribeBridge
	connexion multimédia : DescribeFlow
	connexion multimédia : DescribeFlow SourceMetadata
	connexion multimédia : DescribeGateway
	mediaconnect : Instance DescribeGateway
	connexion multimédia : DescribeOffering
	connexion multimédia : DescribeReservation
	mediaconnect : Droits GrantFlow

Préfixe de service	Actions
	connexion multimédia : ListBridges
	connexion multimédia : ListEntitlements
	connexion multimédia : ListFlows
	mediaconnect : Instances ListGateway
	connexion multimédia : ListGateways
	connexion multimédia : ListOfferings
	connexion multimédia : ListReservations
	connexion multimédia : PurchaseOffering
	mediaconnect : Sortie RemoveBridge
	mediaconnect : Source RemoveBridge
	connexion multimédia : RemoveFlow MediaStream
	mediaconnect : Sortie RemoveFlow
	mediaconnect : Source RemoveFlow
	connexion multimédia : RemoveFlow VpcInterface
	mediaconnect : Autorisation RevokeFlow
	connexion multimédia : StartFlow
	connexion multimédia : StopFlow
	connexion multimédia : UpdateBridge
	mediaconnect : Sortie UpdateBridge
	mediaconnect : Source UpdateBridge
	mediaconnect : État UpdateBridge

Préfixe de service	Actions
	connexion multimédia : UpdateFlow mediaconnect : Autorisation UpdateFlow connexion multimédia : UpdateFlow MediaStream mediaconnect : Sortie UpdateFlow mediaconnect : Source UpdateFlow mediaconnect : Instance UpdateGateway

Préfixe de service	Actions
mediaconvert	conversion de média : AssociateCertificate conversion de média : CancelJob conversion de média : CreateJob mediaconvert : Modèle CreateJob conversion de média : CreatePreset conversion de média : CreateQueue mediaconvert : Modèle DeleteJob conversion de média : DeletePolicy conversion de média : DeletePreset conversion de média : DeleteQueue conversion de média : DescribeEndpoints conversion de média : DisassociateCertificate conversion de média : GetJob mediaconvert : Modèle GetJob conversion de média : GetPolicy conversion de média : GetPreset conversion de média : GetQueue conversion de média : ListJobs mediaconvert : Modèles ListJob conversion de média : ListPresets conversion de média : ListQueues

Préfixe de service	Actions
	conversion de média : PutPolicy mediaconvert : Modèle UpdateJob conversion de média : UpdatePreset conversion de média : UpdateQueue

Préfixe de service	Actions
medialive	MediaLive : AcceptInput DeviceTransfer
	MediaLive : BatchDelete
	MediaLive : BatchStart
	MediaLive : BatchStop
	medialive : Agenda BatchUpdate
	MediaLive : CancelInput DeviceTransfer
	MediaLive : ClaimDevice
	MediaLive : CreateChannel
	medialive : Modèle CreateCloud WatchAlarm
	MediaLive : CreateCloud WatchAlarm TemplateGroup
	medialive : Modèle CreateEvent BridgeRule
	MediaLive : CreateEvent BridgeRule TemplateGroup
	MediaLive : CreateInput
	MediaLive : CreateInput SecurityGroup
	MediaLive : CreateMultiplex
	medialive : Programme CreateMultiplex
	medialive : Entrée CreatePartner
	medialive : Carte CreateSignal
	MediaLive : DeleteChannel
	medialive : Modèle DeleteCloud WatchAlarm
	MediaLive : DeleteCloud WatchAlarm TemplateGroup

Préfixe de service	Actions
	medialive : Modèle DeleteEvent BridgeRule
	MediaLive : DeleteEvent BridgeRule TemplateGroup
	MediaLive : DeleteInput
	MediaLive : DeleteInput SecurityGroup
	MediaLive : DeleteMultiplex
	medialive : Programme DeleteMultiplex
	MediaLive : DeleteReservation
	MediaLive : DeleteSchedule
	medialive : Carte DeleteSignal
	medialive : Configuration DescribeAccount
	MediaLive : DescribeChannel
	MediaLive : DescribeInput
	medialive : Appareil DescribeInput
	MediaLive : DescribeInput DeviceThumbnail
	MediaLive : DescribeInput SecurityGroup
	MediaLive : DescribeMultiplex
	medialive : Programme DescribeMultiplex
	MediaLive : DescribeOffering
	MediaLive : DescribeReservation
	MediaLive : DescribeSchedule
	MediaLive : DescribeThumbnails

Préfixe de service	Actions
	medialive : Modèle GetCloud WatchAlarm
	MediaLive : GetCloud WatchAlarm TemplateGroup
	medialive : Modèle GetEvent BridgeRule
	MediaLive : GetEvent BridgeRule TemplateGroup
	medialive : Carte GetSignal
	MediaLive : ListChannels
	MediaLive : ListCloud WatchAlarm TemplateGroups
	medialive : Modèles ListCloud WatchAlarm
	MediaLive : ListEvent BridgeRule TemplateGroups
	medialive : Modèles ListEvent BridgeRule
	medialive : Appareils ListInput
	MediaLive : ListInput DeviceTransfers
	MediaLive : ListInputs
	MediaLive : ListInput SecurityGroups
	MediaLive : ListMultiplexes
	medialive : Programmes ListMultiplex
	MediaLive : ListOfferings
	MediaLive : ListReservations
	medialive : Cartes ListSignal
	MediaLive : PurchaseOffering
	medialive : Appareil RebootInput

Préfixe de service	Actions
	MediaLive : RejectInput DeviceTransfer
	medialive : Canalisations RestartChannel
	MediaLive : StartChannel
	MediaLive : StartDelete MonitorDeployment
	medialive : Appareil StartInput
	medialive : Fenêtre StartInput DeviceMaintenance
	medialive : Déploiement StartMonitor
	MediaLive : StartMultiplex
	MediaLive : StartUpdate SignalMap
	MediaLive : StopChannel
	medialive : Appareil StopInput
	MediaLive : StopMultiplex
	medialive : Appareil TransferInput
	medialive : Configuration UpdateAccount
	MediaLive : UpdateChannel
	medialive : Classe UpdateChannel
	medialive : Modèle UpdateCloud WatchAlarm
	MediaLive : UpdateCloud WatchAlarm TemplateGroup
	medialive : Modèle UpdateEvent BridgeRule
	MediaLive : UpdateEvent BridgeRule TemplateGroup
	MediaLive : UpdateInput

Préfixe de service	Actions
	medialive : Appareil UpdateInput MediaLive : UpdateInput SecurityGroup MediaLive : UpdateMultiplex medialive : Programme UpdateMultiplex MediaLive : UpdateReservation

Préfixe de service	Actions
mediapackage	package multimédia : ConfigureLogs package multimédia : CreateChannel paquet média : Job CreateHarvest package multimédia : Endpoint CreateOrigin package multimédia : DeleteChannel package multimédia : Endpoint DeleteOrigin package multimédia : DescribeChannel paquet média : Job DescribeHarvest package multimédia : Endpoint DescribeOrigin package multimédia : ListChannels mediapackage : Offres d'emploi ListHarvest mediapackage : Endpoints ListOrigin package multimédia : ListTags ForResource mediapackage : Informations d'identification RotateChannel package multimédia : RotateIngest EndpointCredentials package multimédia : TagResource package multimédia : UntagResource package multimédia : UpdateChannel package multimédia : Endpoint UpdateOrigin

Préfixe de service	Actions
mediapackage-vod	mediapackage-vod : ConfigureLogs mediapackage-vod : CreateAsset mediapackage-vod : Configuration CreatePackaging mediapackage-vod : Groupe CreatePackaging mediapackage-vod : DeleteAsset mediapackage-vod : Configuration DeletePackaging mediapackage-vod : Groupe DeletePackaging mediapackage-vod : DescribeAsset mediapackage-vod : Configuration DescribePackaging mediapackage-vod : Groupe DescribePackaging mediapackage-vod : ListAssets mediapackage-vod : Configurations ListPackaging mediapackage-vod : Groupes ListPackaging mediapackage-vod : ListTags ForResource mediapackage-vod : TagResource mediapackage-vod : UntagResource mediapackage-vod : Groupe UpdatePackaging

Préfixe de service	Actions
mediastore	magasin de médias : CreateContainer magasin de médias : DeleteContainer mediastore : Politique DeleteContainer mediastore : Politique DeleteCors mediastore : Politique DeleteLifecycle mediastore : Politique DeleteMetric magasin de médias : DescribeContainer mediastore : Politique GetContainer mediastore : Politique GetCors mediastore : Politique GetLifecycle mediastore : Politique GetMetric magasin de médias : ListContainers mediastore : Politique PutContainer mediastore : Politique PutCors mediastore : Politique PutLifecycle mediastore : Politique PutMetric mediastore : Journalisation StartAccess mediastore : Journalisation StopAccess

Préfixe de service	Actions
mediatailor	mediatailor : Configuration ConfigureLogs ForPlayback tailleur multimédia : CreateChannel mediatailor : Source CreateLive mediatailor : Calendrier CreatePrefetch tailleur multimédia : CreateProgram mediatailor : Localisation CreateSource mediatailor : Source CreateVod tailleur multimédia : DeleteChannel mediatailor : Politique DeleteChannel mediatailor : Source DeleteLive mediatailor : Configuration DeletePlayback mediatailor : Calendrier DeletePrefetch tailleur multimédia : DeleteProgram mediatailor : Localisation DeleteSource mediatailor : Source DeleteVod tailleur multimédia : DescribeChannel mediatailor : Source DescribeLive tailleur multimédia : DescribeProgram mediatailor : Localisation DescribeSource mediatailor : Source DescribeVod mediatailor : Politique GetChannel

Préfixe de service	Actions
	mediatailor : Calendrier GetChannel
	mediatailor : Configuration GetPlayback
	mediatailor : Calendrier GetPrefetch
	tailleur multimédia : ListAlerts
	tailleur multimédia : ListChannels
	mediatailor : Sources ListLive
	mediatailor : Configurations ListPlayback
	mediatailor : Horaires ListPrefetch
	mediatailor : Localisations ListSource
	mediatailor : Sources ListVod
	mediatailor : Politique PutChannel
	mediatailor : Configuration PutPlayback
	tailleur multimédia : StartChannel
	tailleur multimédia : StopChannel
	tailleur multimédia : UpdateChannel
	mediatailor : Source UpdateLive
	tailleur multimédia : UpdateProgram
	mediatailor : Localisation UpdateSource
	mediatailor : Source UpdateVod

Préfixe de service	Actions
memorydb	memorydb : Cluster BatchUpdate base de données de mémoire : CopySnapshot base de données de mémoire : CreateAcl base de données de mémoire : CreateCluster memorydb : Groupe CreateParameter base de données de mémoire : CreateSnapshot memorydb : Groupe CreateSubnet base de données de mémoire : CreateUser base de données de mémoire : DeleteAcl base de données de mémoire : DeleteCluster memorydb : Groupe DeleteParameter base de données de mémoire : DeleteSnapshot memorydb : Groupe DeleteSubnet base de données de mémoire : DeleteUser base de données de mémoire : DescribeAcls base de données de mémoire : DescribeClusters memorydb : Versions DescribeEngine base de données de mémoire : DescribeEvents memorydb : Groupes DescribeParameter base de données de mémoire : DescribeParameters memorydb : Nœuds DescribeReserved

Préfixe de service	Actions
	base de données de mémoire : DescribeReserved NodesOfferings memorydb : Mises à jour DescribeService base de données de mémoire : DescribeSnapshots memorydb : Groupes DescribeSubnet base de données de mémoire : DescribeUsers base de données de mémoire : FailoverShard memorydb : Mises à jour ListAllowed NodeType base de données de mémoire : PurchaseReserved NodesOffering memorydb : Groupe ResetParameter base de données de mémoire : UpdateAcl base de données de mémoire : UpdateCluster memorydb : Groupe UpdateParameter memorydb : Groupe UpdateSubnet base de données de mémoire : UpdateUser

Préfixe de service	Actions
mgh	mgh : Artifact AssociateCreated mgh : Ressource AssociateDiscovered mgh : CreateHome RegionControl mgh : CreateProgress UpdateStream mgh : DeleteHome RegionControl mgh : DeleteProgress UpdateStream mgh : État DescribeApplication mgh : DescribeHome RegionControls mgh : Tâche DescribeMigration mgh : Artifact DisassociateCreated mgh : Ressource DisassociateDiscovered mgh : Région GetHome mgh : Tâche ImportMigration mgh : États ListApplication mgh : Artefacts ListCreated mgh : Ressources ListDiscovered mgh : Tâches ListMigration mgh : ListProgress UpdateStreams mgh : État NotifyApplication mgh : NotifyMigration TaskState mgh : Attributs PutResource

Préfixe de service	Actions
mgn	mitrailleuse : ArchiveApplication mitrailleuse : ArchiveWave mitrailleuse : AssociateApplications mgn : Serveurs AssociateSource mgn : État ChangeServer LifeCycle mitrailleuse : CreateApplication mitrailleuse : CreateConnector mitrailleuse : CreateLaunch ConfigurationTemplate mitrailleuse : CreateReplication ConfigurationTemplate mitrailleuse : CreateWave mitrailleuse : DeleteApplication mitrailleuse : DeleteConnector mitrailleuse : DeleteJob mitrailleuse : DeleteLaunch ConfigurationTemplate mitrailleuse : DeleteReplication ConfigurationTemplate mgn : Serveur DeleteSource mgn : Client DeleteVcenter mitrailleuse : DeleteWave mitrailleuse : DescribeJob LogItems mitrailleuse : DescribeJobs mitrailleuse : DescribeLaunch ConfigurationTemplates

Préfixe de service	Actions
	mitrailleuse : DescribeReplication ConfigurationTemplates mgn : Clients DescribeVcenter mitrailleuse : DisassociateApplications mgn : Serveurs DisassociateSource mgn : Service DisconnectFrom mitrailleuse : FinalizeCutover mgn : Configuration GetReplication mitrailleuse : InitializeService mitrailleuse : ListConnectors mgn : Erreurs ListExport mitrailleuse : ListExports mgn : Erreurs ListImport mitrailleuse : ListImports mgn : Comptes ListManaged mitrailleuse : ListSource ServerActions mgn : Actions ListTemplate mgn : Archivé MarkAs mitrailleuse : PauseReplication mitrailleuse : PutSource ServerAction mgn : Action PutTemplate mitrailleuse : RemoveSource ServerAction

Préfixe de service	Actions
	mgn : Action RemoveTemplate mitrailleuse : ResumeReplication mgn : Réplication RetryData mitrailleuse : StartCutover mitrailleuse : StartExport mitrailleuse : StartImport mitrailleuse : StartReplication mitrailleuse : StartTest mitrailleuse : StopReplication mgn : Instances TerminateTarget mitrailleuse : UnarchiveApplication mitrailleuse : UnarchiveWave mitrailleuse : UpdateApplication mitrailleuse : UpdateConnector mitrailleuse : UpdateLaunch ConfigurationTemplate mgn : Configuration UpdateReplication mitrailleuse : UpdateReplication ConfigurationTemplate mgn : Serveur UpdateSource mgn : Type UpdateSource ServerReplication mitrailleuse : UpdateWave

Préfixe de service	Actions
migrationhub-strategy	migrationhub-strategy : modèle GetAnti stratégie du pôle migratoire : GetApplication ComponentDetails stratégie du pôle migratoire : GetApplication ComponentStrategies stratégie du pôle migratoire : GetAssessment stratégie du pôle migratoire : GetImport FileTask stratégie du pôle migratoire : GetLatest AssessmentId stratégie du pôle migratoire : GetMessage migrationhub-strategy : Préférences GetPortfolio migrationhub-strategy : résumé GetPortfolio stratégie du pôle migratoire : GetRecommendation ReportDetails migrationhub-strategy : Détails GetServer migrationhub-strategy : Stratégies GetServer migrationhub-strategy : Serveurs ListAnalyzable migrationhub-strategy : modèles ListAnti migrationhub-strategy : Composants ListApplication stratégie du pôle migratoire : ListCollectors stratégie du pôle migratoire : ListImport FileTask migrationhub-strategy : Artefacts ListJar stratégie du pôle migratoire : ListServers migrationhub-strategy : Préférences PutPortfolio stratégie du pôle migratoire : RegisterCollector

Préfixe de service	Actions
	stratégie du pôle migratoire : SendMessage
	stratégie du pôle migratoire : StartAssessment
	stratégie du pôle migratoire : StartImport FileTask
	stratégie du pôle migratoire : StartRecommendation ReportGen eration
	stratégie du pôle migratoire : StopAssessment
	stratégie du pôle migratoire : UpdateApplication ComponentConfig
	migrationhub-strategy : Configuration UpdateCollector
	migrationhub-strategy : Config UpdateServer

Préfixe de service	Actions
mobiletargeting	ciblage mobile : CreateApp ciblage mobile : CreateCampaign ciblage mobile : modèle CreateEmail ciblage mobile : Job CreateExport ciblage mobile : Job CreateImport ciblage mobile : CreateIn AppTemplate ciblage mobile : CreateJourney ciblage mobile : modèle CreatePush ciblage mobile : Configuration CreateRecommender ciblage mobile : CreateSegment ciblage mobile : modèle CreateSms ciblage mobile : modèle CreateVoice ciblage mobile : Canal DeleteAdm ciblage mobile : Canal DeleteApns ciblage mobile : DeleteApns SandboxChannel ciblage mobile : DeleteApns VoipChannel ciblage mobile : Canal DeleteApns VoipSandbox ciblage mobile : DeleteApp ciblage mobile : Canal DeleteBaidu ciblage mobile : DeleteCampaign ciblage mobile : Canal DeleteEmail

Préfixe de service	Actions
	ciblage mobile : modèle DeleteEmail ciblage mobile : DeleteEndpoint ciblage mobile : Stream DeleteEvent ciblage mobile : Canal DeleteGcm ciblage mobile : DeleteIn AppTemplate ciblage mobile : DeleteJourney ciblage mobile : modèle DeletePush ciblage mobile : Configuration DeleteRecommender ciblage mobile : DeleteSegment ciblage mobile : Canal DeleteSms ciblage mobile : modèle DeleteSms ciblage mobile : points de terminaison DeleteUser ciblage mobile : Canal DeleteVoice ciblage mobile : modèle DeleteVoice ciblage mobile : Canal GetAdm ciblage mobile : Canal GetApns ciblage mobile : GetApns SandboxChannel ciblage mobile : GetApns VoipChannel ciblage mobile : Canal GetApns VoipSandbox ciblage mobile : GetApp ciblage mobile : Kip GetApplication DateRange

Préfixe de service	Actions
	ciblage mobile : Paramètres GetApplication ciblage mobile : GetApps ciblage mobile : Canal GetBaidu ciblage mobile : GetCampaign ciblage mobile : activités GetCampaign ciblage mobile : Kip GetCampaign DateRange ciblage mobile : GetCampaigns ciblage mobile : Version GetCampaign ciblage mobile : versions GetCampaign ciblage mobile : GetChannels ciblage mobile : Canal GetEmail ciblage mobile : modèle GetEmail ciblage mobile : GetEndpoint ciblage mobile : Stream GetEvent ciblage mobile : Job GetExport ciblage mobile : Offres d'emploi GetExport ciblage mobile : Canal GetGcm ciblage mobile : Job GetImport ciblage mobile : Offres d'emploi GetImport ciblage mobile : GetIn AppMessages ciblage mobile : GetIn AppTemplate

Préfixe de service	Actions
	ciblage mobile : GetJourney ciblage mobile : Kip GetJourney DateRange ciblage mobile : métriques GetJourney ExecutionActivity ciblage mobile : GetJourney ExecutionMetrics ciblage mobile : GetJourney RunExecution ActivityMetrics ciblage mobile : métriques GetJourney RunExecution ciblage mobile : Exécute GetJourney ciblage mobile : modèle GetPush ciblage mobile : Configuration GetRecommender ciblage mobile : configurations GetRecommender ciblage mobile : GetSegment ciblage mobile : GetSegment ExportJobs ciblage mobile : GetSegment ImportJobs ciblage mobile : GetSegments ciblage mobile : Version GetSegment ciblage mobile : versions GetSegment ciblage mobile : Canal GetSms ciblage mobile : modèle GetSms ciblage mobile : points de terminaison GetUser ciblage mobile : Canal GetVoice ciblage mobile : modèle GetVoice

Préfixe de service	Actions
	ciblage mobile : ListJourneys ciblage mobile : ListTemplates ciblage mobile : versions ListTemplate ciblage mobile : Valider PhoneNumber ciblage mobile : Stream PutEvent ciblage mobile : RemoveAttributes ciblage mobile : Canal UpdateAdm ciblage mobile : Canal UpdateApns ciblage mobile : UpdateApns SandboxChannel ciblage mobile : UpdateApns VoipChannel ciblage mobile : Canal UpdateApns VoipSandbox ciblage mobile : Paramètres UpdateApplication ciblage mobile : Canal UpdateBaidu ciblage mobile : UpdateCampaign ciblage mobile : Canal UpdateEmail ciblage mobile : modèle UpdateEmail ciblage mobile : UpdateEndpoint ciblage mobile : Batch UpdateEndpoints ciblage mobile : Canal UpdateGcm ciblage mobile : UpdateIn AppTemplate ciblage mobile : UpdateJourney

Préfixe de service	Actions
	ciblage mobile : État UpdateJourney ciblage mobile : modèle UpdatePush ciblage mobile : Configuration UpdateRecommender ciblage mobile : UpdateSegment ciblage mobile : Canal UpdateSms ciblage mobile : modèle UpdateSms ciblage mobile : UpdateTemplate ActiveVersion ciblage mobile : Canal UpdateVoice ciblage mobile : modèle UpdateVoice mobiletargeting:VerifyOTPMessage

Préfixe de service	Actions
mq	mq : CreateBroker mq : CreateConfiguration mq : CreateUser mq : DeleteBroker mq : DeleteUser mq : DescribeBroker mq : DescribeBroker EngineTypes mq : DescribeBroker InstanceOptions mq : DescribeConfiguration mq : Révision DescribeConfiguration mq : DescribeUser mq : ListBrokers mq : Révisions ListConfiguration mq : ListConfigurations mq : ListUsers mq:Promote mq : RebootBroker mq : UpdateBroker mq : UpdateConfiguration mq : UpdateUser

Préfixe de service	Actions
networkmanager	gestionnaire de réseau : AcceptAttachment gestionnaire de réseau : Peer AssociateConnect gestionnaire de réseau : Gateway AssociateCustomer gestionnaire de réseau : AssociateLink gestionnaire de réseau : Peer AssociateTransit GatewayConnect networkmanager : pièce jointe CreateConnect gestionnaire de réseau : CreateConnection gestionnaire de réseau : Peer CreateConnect networkmanager : CreateCore Network gestionnaire de réseau : CreateDevice networkmanager : CreateGlobal Network gestionnaire de réseau : CreateLink gestionnaire de réseau : CreateSite gestionnaire de réseau : CreateSite ToSite VpnAttachment gestionnaire de réseau : CreateTransit GatewayPeering gestionnaire de réseau : CreateTransit GatewayRoute TableAttachment networkmanager : pièce jointe CreateVpc gestionnaire de réseau : DeleteAttachment gestionnaire de réseau : DeleteConnection gestionnaire de réseau : Peer DeleteConnect

Préfixe de service	Actions
	networkmanager : DeleteCore Network gestionnaire de réseau : Version DeleteCore NetworkPolicy gestionnaire de réseau : DeleteDevice networkmanager : DeleteGlobal Network gestionnaire de réseau : DeleteLink gestionnaire de réseau : DeletePeering networkmanager : Politique DeleteResource gestionnaire de réseau : DeleteSite gestionnaire de réseau : Gateway DeregisterTransit networkmanager : DescribeGlobal Réseaux gestionnaire de réseau : Peer DisassociateConnect gestionnaire de réseau : Gateway DisassociateCustomer gestionnaire de réseau : DisassociateLink gestionnaire de réseau : Peer DisassociateTransit GatewayConnect gestionnaire de réseau : Set ExecuteCore NetworkChange networkmanager : pièce jointe GetConnect gestionnaire de réseau : GetConnections gestionnaire de réseau : Peer GetConnect gestionnaire de réseau : GetConnect PeerAssociations networkmanager : GetCore Network networkmanager : Evénements GetCore NetworkChange

Préfixe de service	Actions
	gestionnaire de réseau : Set GetCore NetworkChange gestionnaire de réseau : GetCore NetworkPolicy gestionnaire de réseau : GetCustomer GatewayAssociations gestionnaire de réseau : GetDevices gestionnaire de réseau : Associations GetLink gestionnaire de réseau : GetLinks gestionnaire de réseau : GetNetwork ResourceCounts gestionnaire de réseau : GetNetwork ResourceRelationships networkmanager : Ressources GetNetwork gestionnaire de réseau : Routes GetNetwork networkmanager : Télémétrie GetNetwork networkmanager : Politique GetResource networkmanager : Analyse GetRoute gestionnaire de réseau : GetSites gestionnaire de réseau : GetSite ToSite VpnAttachment gestionnaire de réseau : GetTransit GatewayConnect PeerAssoc iations gestionnaire de réseau : GetTransit GatewayPeering gestionnaire de réseau : GetTransit GatewayRegistrations gestionnaire de réseau : GetTransit GatewayRoute TableAtta chment networkmanager : pièce jointe GetVpc

Préfixe de service	Actions
	gestionnaire de réseau : ListAttachments gestionnaire de réseau : Peers ListConnect gestionnaire de réseau : Versions ListCore NetworkPolicy networkmanager : ListCore Réseaux networkmanager : État ListOrganization ServiceAccess gestionnaire de réseau : ListPeerings gestionnaire de réseau : PutCore NetworkPolicy networkmanager : Politique PutResource gestionnaire de réseau : Gateway RegisterTransit gestionnaire de réseau : RejectAttachment gestionnaire de réseau : Version RestoreCore NetworkPolicy networkmanager : StartOrganization ServiceAccess Mise à jour networkmanager : Analyse StartRoute gestionnaire de réseau : UpdateConnection networkmanager : UpdateCore Network gestionnaire de réseau : UpdateDevice networkmanager : UpdateGlobal Network gestionnaire de réseau : UpdateLink gestionnaire de réseau : UpdateNetwork ResourceMetadata gestionnaire de réseau : UpdateSite networkmanager : pièce jointe UpdateVpc

Préfixe de service	Actions
nimble	agile : AcceptEulas nimble : Profil CreateLaunch agile : Image CreateStreaming agile : Session CreateStreaming agile : CreateStreaming SessionStream agile : CreateStudio agile : Composant CreateStudio nimble : Profil DeleteLaunch agile : DeleteLaunch ProfileMember agile : Image DeleteStreaming agile : Session DeleteStreaming agile : DeleteStudio agile : Composant DeleteStudio nimble : Membre DeleteStudio agile : GetEula agile : GetLaunch ProfileDetails agile : Image GetStreaming agile : Session GetStreaming agile : GetStreaming SessionBackup agile : GetStreaming SessionStream agile : GetStudio

Préfixe de service	Actions
	agile : Composant GetStudio
	nimble : Membre GetStudio
	agile : ListEulas
	agile : ListLaunch ProfileMembers
	nimble : Profils ListLaunch
	agile : Images ListStreaming
	agile : ListStreaming SessionBackups
	agile : Sessions ListStreaming
	agile : Composants ListStudio
	nimble : Membres ListStudio
	agile : ListStudios
	agile : PutLaunch ProfileMembers
	nimble : Membres PutStudio
	agile : Session StartStreaming
	agile : SSO StartStudio ConfigurationRepair
	agile : Session StopStreaming
	nimble : Profil UpdateLaunch
	agile : UpdateLaunch ProfileMember
	agile : Image UpdateStreaming
	agile : UpdateStudio
	agile : Composant UpdateStudio

Préfixe de service	Actions
omics	omics : AbortMultipart ReadSet Upload bandes dessinées : BatchDelete ReadSet bandes dessinées : CancelAnnotation ImportJob bandes dessinées : CancelRun bandes dessinées : CancelVariant ImportJob omics : CompleteMultipart ReadSet Upload Comics : CreateAnnotation Store omics : CreateMultipart ReadSet Upload Comics : CreateReference Store Comics : CreateRun Groupe Comics : CreateSequence Store Comics : CreateVariant Store bandes dessinées : CreateWorkflow Comics : DeleteAnnotation Store bandes dessinées : DeleteReference Comics : DeleteReference Store bandes dessinées : DeleteRun Comics : DeleteRun Groupe Comics : DeleteSequence Store Comics : DeleteVariant Store bandes dessinées : DeleteWorkflow

Préfixe de service	Actions
	bandes dessinées : GetAnnotation ImportJob Comics : GetAnnotation Store Comics : GetRead Set Comics : GetRead SetActivation Job Comics : GetRead SetExport Job Comics : GetRead SetImport Job bandes dessinées : GetRead SetMetadata bandes dessinées : GetReference bandes dessinées : GetReference ImportJob omics : GetReference métadonnées Comics : GetReference Store bandes dessinées : GetRun Comics : GetRun Groupe omics : GetRun Tâche Comics : GetSequence Store bandes dessinées : GetVariant ImportJob Comics : GetVariant Store bandes dessinées : GetWorkflow bandes dessinées : ListAnnotation ImportJobs Bandes dessinées : ListAnnotation Stories Comics : ListMultipart ReadSet Uploads

Préfixe de service	Actions
	Comics : ListRead SetActivation Offres d'emploi Comics : ListRead SetExport Offres d'emploi Comics : ListRead SetImport Offres d'emploi Comics : ListRead Sets Comics : ListRead SetUpload Pièces bandes dessinées : ListReference ImportJobs bandes dessinées : ListReferences Bandes dessinées : ListReference Stories Comics : ListRun Groupes bandes dessinées : ListRuns Comics : ListRun Tâches Bandes dessinées : ListSequence Stories bandes dessinées : ListVariant ImportJobs Bandes dessinées : ListVariant Stories bandes dessinées : ListWorkflows bandes dessinées : StartAnnotation ImportJob Comics : StartRead SetActivation Job Comics : StartRead SetExport Job Comics : StartRead SetImport Job bandes dessinées : StartReference ImportJob bandes dessinées : StartRun

Préfixe de service	Actions
	bandes dessinées : StartVariant ImportJob Comics : UpdateAnnotation Store Comics : UpdateRun Groupe Comics : UpdateVariant Store bandes dessinées : UpdateWorkflow bandes dessinées : UploadRead SetPart

Préfixe de service	Actions
opsworks	opsworks : AssignInstance
	opsworks : AssignVolume
	opsworks : IP AssociateElastic
	opsworks : AttachElastic LoadBalancer
	opsworks : CloneStack
	opsworks : CreateApp
	opsworks : CreateDeployment
	opsworks : CreateInstance
	opsworks : CreateLayer
	opsworks : CreateStack
	opsworks : Profil CreateUser
	opsworks : DeleteApp
	opsworks : DeleteInstance
	opsworks : DeleteLayer
	opsworks : DeleteStack
	opsworks : Profil DeleteUser
	opsworks : Cluster DeregisterEcs
	opsworks : IP DeregisterElastic
	opsworks : DeregisterInstance
	opsworks : DeregisterRds DbInstance
	opsworks : DeregisterVolume

Préfixe de service	Actions
	opsworks : Versions DescribeAgent
	opsworks : DescribeApps
	opsworks : DescribeCommands
	opsworks : DescribeDeployments
	opsworks : Clusters DescribeEcs
	opsworks : Ips DescribeElastic
	opsworks : DescribeElastic LoadBalancers
	opsworks : DescribeInstances
	opsworks : DescribeLayers
	opsworks : Mise à l'échelle DescribeLoad BasedAuto
	opsworks : DescribeMy UserProfile
	opsworks : Systèmes DescribeOperating
	opsworks : DescribePermissions
	opsworks : Tableaux DescribeRaid
	opsworks : DescribeRds DbInstances
	opsworks : Erreurs DescribeService
	opsworks : DescribeStack ProvisioningParameters
	opsworks : DescribeStacks
	opsworks : Résumé DescribeStack
	opsworks : Mise à l'échelle DescribeTime BasedAuto
	opsworks : Profils DescribeUser

Préfixe de service	Actions
	opsworks : DescribeVolumes
	opsworks : DetachElastic LoadBalancer
	opsworks : IP DisassociateElastic
	opsworks : Suggestion GetHostname
	opsworks : GrantAccess
	opsworks : RebootInstance
	opsworks : Cluster RegisterEcs
	opsworks : IP RegisterElastic
	opsworks : RegisterInstance
	opsworks : RegisterRds DbInstance
	opsworks : RegisterVolume
	opsworks : Mise à l'échelle SetLoad BasedAuto
	opsworks : SetPermission
	opsworks : Mise à l'échelle SetTime BasedAuto
	opsworks : StartInstance
	opsworks : StartStack
	opsworks : StopInstance
	opsworks : StopStack
	opsworks : UnassignInstance
	opsworks : UnassignVolume
	opsworks : UpdateApp

Préfixe de service	Actions
	opsworks : IP UpdateElastic
	opsworks : UpdateInstance
	opsworks : UpdateLayer
	opsworks : UpdateMy UserProfile
	opsworks : UpdateRds DbInstance
	opsworks : UpdateStack
	opsworks : Profil UpdateUser
	opsworks : UpdateVolume

Préfixe de service	Actions
opsworks-cm	Opsworks-cm : AssociateNode Opsworks-cm : CreateBackup Opsworks-cm : CreateServer Opsworks-cm : DeleteBackup Opsworks-cm : DeleteServer opsworks-cm : Attributs DescribeAccount Opsworks-cm : DescribeBackups Opsworks-cm : DescribeEvents Opsworks-cm : DescribeNode AssociationStatus Opsworks-cm : DescribeServers Opsworks-cm : DisassociateNode Opsworks-cm : ExportServer EngineAttribute Opsworks-cm : RestoreServer Opsworks-cm : StartMaintenance Opsworks-cm : UpdateServer Opsworks-cm : UpdateServer EngineAttributes

Préfixe de service	Actions
organisations	organisations : AcceptHandshake
	organisations : AttachPolicy
	organisations : CancelHandshake
	organisations : CloseAccount
	organisations : CreateAccount
	organisations : CreateGov CloudAccount
	organisations : CreateOrganization
	organisations : CreateOrganizational Unit
	organisations : CreatePolicy
	organisations : DeclineHandshake
	organisations : DeleteOrganization
	organisations : DeleteOrganizational Unit
	organisations : DeletePolicy
	organisations : DeleteResource Politique
	organisations : DeregisterDelegated Administrateur
	organisations : DescribeAccount
	organisations : DescribeCreate AccountStatus
	organisations : DescribeEffective Politique
	organisations : DescribeHandshake
	organisations : DescribeOrganization
	organisations : DescribeOrganizational Unit

Préfixe de service	Actions
	organisations : DescribePolicy
	organisations : DescribeResource Politique
	organisations : DetachPolicy
	Organisations : Disable AWSServiceAccess
	organisations : DisablePolicy Type
	organisations : EnableAll Caractéristiques
	Organisations : activer AWSServiceAccess
	organisations : EnablePolicy Type
	organisations : InviteAccount ToOrganization
	organisations : LeaveOrganization
	organisations : ListAccounts
	organisations : ListAccounts ForParent
	Organisations : liste AWSServiceAccessForOrganization
	organisations : ListChildren
	organisations : ListCreate AccountStatus
	organisations : ListDelegated Administrateurs
	organisations : ListDelegated ServicesFor Compte
	organisations : ListHandshakes ForAccount
	organisations : ListHandshakes ForOrganization
	organisations : ListOrganizational UnitsFor Parent
	organisations : ListParents

Préfixe de service	Actions
	organisations : ListPolicies
	organisations : ListPolicies ForTarget
	organisations : ListRoots
	organisations : ListTargets ForPolicy
	organisations : MoveAccount
	organisations : PutResource Politique
	organisations : RegisterDelegated Administrateur
	organisations : RemoveAccount FromOrganization
	organisations : UpdateOrganizational Unit
	organisations : UpdatePolicy

Préfixe de service	Actions
outposts	avant-postes : CancelCapacity Tâche avant-postes : CancelOrder avant-postes : CreateOrder avant-postes : CreateOutpost avant-postes : CreatePrivate ConnectivityConfig avant-postes : CreateSite avant-postes : DeleteOutpost avant-postes : DeleteSite avant-postes : GetCapacity Tâche avant-postes : Objet GetCatalog avant-postes : GetConnection avant-postes : GetOrder avant-postes : GetOutpost avant-postes : GetOutpost InstanceTypes avant-postes : types GetOutpost SupportedInstance avant-postes : GetPrivate ConnectivityConfig avant-postes : GetSite avant-postes : Adresse GetSite avant-postes : ListAssets avant-postes : tâches ListCapacity avant-postes : objets ListCatalog

Préfixe de service	Actions
	avant-postes : ListOrders avant-postes : ListOutposts avant-postes : ListSites avant-postes : StartCapacity Tâche avant-postes : StartConnection avant-postes : UpdateOutpost avant-postes : UpdateSite avant-postes : Adresse UpdateSite avant-postes : Propriétés UpdateSite RackPhysical

Préfixe de service	Actions
panorama	panorama : CreateApplication Instance
	panorama : CreateJob ForDevices
	panorama : CreateNode FromTemplate Job
	panorama : CreatePackage
	panorama : CreatePackage ImportJob
	panorama : DeleteDevice
	panorama : DeletePackage
	panorama : DeregisterPackage Version
	panorama : DescribeApplication Instance
	panorama : DescribeApplication InstanceDetails
	panorama : DescribeDevice
	panorama : DescribeDevice Job
	panorama : DescribeNode
	panorama : DescribeNode FromTemplate Job
	panorama : DescribePackage
	panorama : DescribePackage ImportJob
	panorama : DescribePackage Version
	panorama : ListApplication InstanceDependencies
	panorama : ListApplication InstanceNode Instances
	panorama : ListApplication Instances
	panorama : ListDevices

Préfixe de service	Actions
	panorama : ListDevices Offres d'emploi panorama : ListNode FromTemplate Offres d'emploi panorama : ListNodes panorama : ListPackage ImportJobs panorama : ListPackages panorama : ProvisionDevice panorama : RegisterPackage Version panorama : RemoveApplication Instance panorama : SignalApplication InstanceNode Instances panorama : UpdateDevice Métadonnées
pi	épi : CreatePerformance AnalysisReport épi : DeletePerformance AnalysisReport pi : DescribeDimension Clés épi : GetDimension KeyDetails épi : GetPerformance AnalysisReport pi : GetResource Métadonnées pi : GetResource Métriques épi : ListAvailable ResourceDimensions épi : ListAvailable ResourceMetrics épi : ListPerformance AnalysisReports

Préfixe de service	Actions
pipes	tuyaux : CreatePipe tuyaux : DeletePipe tuyaux : DescribePipe tuyaux : ListPipes tuyaux : StartPipe tuyaux : StopPipe tuyaux : UpdatePipe
polly	Polly : DeleteLexicon Polly : DescribeVoices Polly : GetLexicon Polly : GetSpeech SynthesisTask Polly : ListLexicons Polly : ListSpeech SynthesisTasks Polly : PutLexicon Polly : StartSpeech SynthesisTask Polly : SynthesizeSpeech

Préfixe de service	Actions
profile	profil : AddProfile Key
	profil : CreateCalculated AttributeDefinition
	profil : CreateDomain
	profil : CreateEvent Stream
	profil : CreateProfile
	profil : DeleteCalculated AttributeDefinition
	profil : DeleteDomain
	profil : DeleteEvent Stream
	profil : DeleteIntegration
	profil : DeleteProfile
	profil : DeleteProfile Key
	profil : DeleteProfile Object
	profil : DeleteProfile ObjectType
	profil : DeleteWorkflow
	profil : DetectProfile ObjectType
	profil : GetAuto MergingPreview
	profil : GetCalculated AttributeDefinition
	profil : GetCalculated AttributeFor Profil
	profil : GetDomain
	profil : GetEvent Stream
	profil : GetIdentity ResolutionJob

Préfixe de service	Actions
	profil : GetIntegration
	profil : GetMatches
	profil : GetProfile ObjectType
	profil : GetProfile ObjectType Template
	profil : GetSimilar Profiles
	profil : GetWorkflow
	profil : GetWorkflow Steps
	profil : ListAccount Integrations
	profil : ListCalculated AttributeDefinitions
	profil : ListCalculated AttributesFor Profil
	profil : ListDomains
	profil : ListEvent Streams
	profil : ListIdentity ResolutionJobs
	profil : ListIntegrations
	profil : ListProfile Objects
	profil : ListProfile ObjectTypes
	profil : ListProfile ObjectType Modèles
	profil : ListRule BasedMatches
	profil : ListWorkflows
	profil : MergeProfiles
	profil : PutIntegration

Préfixe de service	Actions
	profil : PutProfile Object profil : PutProfile ObjectType profil : SearchProfiles profil : UpdateCalculated AttributeDefinition profil : UpdateDomain profil : UpdateProfile

Préfixe de service	Actions
qldb	qldb : CancelJournal KinesisStream qldb : CreateLedger qldb : DeleteLedger qldb : DescribeJournal KinesisStream qldb : Export S3 DescribeJournal qldb : DescribeLedger qldb : ToS3 ExportJournal qldb : GetBlock qldb : GetDigest qldb : GetRevision qldb : ListJournal KinesisStreams ForLedger qldb : Exportations S3 ListJournal qldb : ListJournal Ledger S3 ExportsFor qldb : ListLedgers qldb : StreamJournal ToKinesis qldb : UpdateLedger qldb : UpdateLedger PermissionsMode

Préfixe de service	Actions
ram	RAM : AcceptResource ShareInvitation
	RAM : AssociateResource Partager
	RAM : AssociateResource SharePermission
	RAM : CreatePermission
	RAM : CreatePermission Version
	RAM : CreateResource Partager
	RAM : DeletePermission
	RAM : DeletePermission Version
	RAM : DeleteResource Partager
	RAM : DisassociateResource Partager
	RAM : DisassociateResource SharePermission
	Gram : EnableSharing WithAws Organisation
	RAM : GetPermission
	ram : GetResource Politiques
	RAM : GetResource ShareAssociations
	RAM : GetResource ShareInvitations
	RAM : GetResource Shares
	RAM : ListPending InvitationResources
	RAM : ListPermission Associations
	RAM : ListPermissions
	RAM : ListPermission Versions

Préfixe de service	Actions
	RAM : ListPrincipals RAM : ListReplace PermissionAssociations Work RAM : ListResources RAM : ListResource SharePermissions RAM : ListResource types ram : PromotePermission CreatedFrom Politique RAM : PromoteResource ShareCreated FromPolicy RAM : RejectResource ShareInvitation RAM : ReplacePermission Associations RAM : SetDefault PermissionVersion RAM : UpdateResource Partager
rbin	corbeille : CreateRule corbeille : DeleteRule corbeille : GetRule corbeille : ListRules corbeille : LockRule corbeille : UnlockRule corbeille : UpdateRule

Préfixe de service	Actions
rds	Réseau : TodbCluster AddRole rds : TODBInstance AddRole rds : Abonnement AddSource IdentifierTo rouge : ApplyPending MaintenanceAction RDS SecurityGroup : entrée de base de données autorisée rds:BacktrackDBCluster rds : Tâche CancelExport RDS:CopyDB Group ClusterParameter RDS : CopyDB ClusterSnapshot RDS : CopyDB ParameterGroup rds:CopyDBSnapshot Rds : Groupe CopyOption rouge : DB CreateCustom EngineVersion RDS ClusterParameter : CreateDB Group RDS : CreateDB ClusterSnapshot RDS : CreateDB ParameterGroup rds:CreateDBProxy RDS : CreateDB ProxyEndpoint RDS : CreateDB SecurityGroup rds:CreateDBSnapshot RDS : CreateDB SubnetGroup

Préfixe de service	Actions
	rds : Abonnement CreateEvent
	rds : Cluster CreateGlobal
	Rds : Groupe CreateOption
	rouge : DeleteBlue GreenDeployment
	RDS : DeleteDB Backup ClusterAutomated
	RDS ClusterParameter : groupe de base de données supprimé
	RDS : DeleteDB ClusterSnapshot
	RDS : DeleteDB Backup InstanceAutomated
	RDS : DeleteDB ParameterGroup
	rds:DeleteDBProxy
	RDS : DeleteDB ProxyEndpoint
	RDS : DeleteDB SecurityGroup
	rds:DeleteDBSnapshot
	RDS : DeleteDB SubnetGroup
	rds : Abonnement DeleteEvent
	rds : Cluster DeleteGlobal
	Rds : Groupe DeleteOption
	RDS : Désenregistrer la base de données ProxyTargets
	rds : Attributs DescribeAccount
	rouge : DescribeBlue GreenDeployments
	rouge : DescribeCertificates

Préfixe de service	Actions
	RDS ClusterAutomated : sauvegardes de base de données décrites
	RDS : décrit B ClusterBacktracks
	RDS : décrit B ClusterEndpoints
	RDS ClusterParameter : groupes B décrits
	RDS : décrit B ClusterParameters
	rds:DescribeDBClusters
	RDS:Attributs DescribeDB ClusterSnapshot
	RDS : décrit B ClusterSnapshots
	RDS : décrit B EngineVersions
	RDS InstanceAutomated : sauvegardes de base de données décrites
	rds:DescribeDBInstances
	RDS : décrit B LogFiles
	RDS : décrit B ParameterGroups
	rds:DescribeDBParameters
	rds:DescribeDBProxies
	RDS : décrit B ProxyEndpoints
	RDS ProxyTarget : groupes B décrits
	RDS : décrit B ProxyTargets
	RDS : recommandations décrites
	RDS : décrit B SecurityGroups

Préfixe de service	Actions
	RDS : décrit B SnapshotAttributes
	rds:DescribeDBSnapshots
	rds : Bases de données DescribeDb SnapshotTenant
	RDS : décrit B SubnetGroups
	rds : Paramètres DescribeEngine DefaultCluster
	rouge : DescribeEngine DefaultParameters
	RDS : Catégories DescribeEvent
	rouge : DescribeEvents
	rds : Abonnements DescribeEvent
	rds : Tâches DescribeExport
	RDS : Clusters DescribeGlobal
	rouge : DescribeIntegrations
	rouge : DescribeOption GroupOptions
	Rds : Groupes DescribeOption
	rouge : DB DescribeOrderable InstanceOptions
	rouge : DescribePending MaintenanceActions
	rds : DBInstances DescribeReserved
	rouge : DB DescribeReserved InstancesOfferings
	Rds : Régions DescribeSource
	rds : Bases de données DescribeTenant
	rouge : DB DescribeValid InstanceModifications

Préfixe de service	Actions
	rouge : DB DownloadComplete LogFile RDS LogFile : Télécharger la partie DB rds:FailoverDBCluster rds : Cluster FailoverGlobal rds : Stream ModifyActivity rouge : ModifyCertificates rouge : DB ModifyCurrent ClusterCapacity RDS : Modifier la base de données ClusterEndpoint RDS ClusterParameter : Modifier le groupe de base de données Attribut RDS:ModifyDB ClusterSnapshot RDS : Modifier la base de données ParameterGroup rds:ModifyDBProxy RDS : Modifier la base de données ProxyEndpoint RDS ProxyTarget : Modifier le groupe de base de données RDS : Modifier la recommandation DB rds:ModifyDBSnapshot RDS : Modifier la base de données SnapshotAttribute RDS : Modifier la base de données SubnetGroup rds : Abonnement ModifyEvent rds : Cluster ModifyGlobal Rds : Groupe ModifyOption

Préfixe de service	Actions
	rds : Base de données ModifyTenant
	rouge : DB PurchaseReserved InstancesOffering
	rds:RebootDBCluster
	RDS : RegisterDB ProxyTargets
	rouge : RemoveFrom GlobalCluster
	rds : FromDBCluster RemoveRole
	rds : FromDBInstance RemoveRole
	rds : Abonnement RemoveSource IdentifierFrom
	RDS ClusterParameter : ResetDB Group
	RDS : ResetDB ParameterGroup
	RDS : RestoreDB S3 ClusterFrom
	RDS ClusterFrom : RestoreDB Snapshot
	RDS ClusterTo PointIn : RestoreDB Time
	RDS InstanceFrom : RestoreDB DB Snapshot
	RDS : RestoreDB S3 InstanceFrom
	RDS InstanceTo PointIn : RestoreDB Time
	RDS SecurityGroup : RevokeDB Ingress
	rds : Stream StartActivity
	rds:StartDBCluster
	rds:StartDBInstance
	RDS : StartDB InstanceAutomated BackupsReplication

Préfixe de service	Actions
	rds : Tâche StartExport rds : Stream StopActivity rds:StopDBCluster rds:StopDBInstance RDS : StopDB InstanceAutomated BackupsReplication rouge : SwitchoverBlue GreenDeployment rds : Cluster SwitchoverGlobal RDS : Réplique SwitchoverRead

Préfixe de service	Actions
redshift	décalage vers le rouge : AcceptReserved NodeExchange décalage vers le rouge : AddPartner décalage vers le rouge : AssociateData ShareConsumer redshift : Ingress AuthorizeCluster SecurityGroup redshift : Partager AuthorizeData redshift : Accès AuthorizeEndpoint redshift : Accès AuthorizeSnapshot décalage vers le rouge : BatchDelete ClusterSnapshots décalage vers le rouge : BatchModify ClusterSnapshots décalage vers le rouge : CancelResize redshift : CopyCluster Instantané redshift : Profil CreateAuthentication décalage vers le rouge : CreateCluster décalage vers le rouge : CreateCluster ParameterGroup décalage vers le rouge : CreateCluster SecurityGroup redshift : CreateCluster Instantané décalage vers le rouge : CreateCluster SubnetGroup décalage vers le rouge : CreateCustom DomainAssociation redshift : Accès CreateEndpoint redshift : Abonnement CreateEvent décalage vers le rouge : CreateHsm ClientCertificate

Préfixe de service	Actions
	redshift : Configuration CreateHsm décalage vers le rouge : CreateRedshift IdcApplication redshift : Action CreateScheduled décalage vers le rouge : CreateSnapshot CopyGrant redshift : Calendrier CreateSnapshot redshift : limite CreateUsage redshift : Partager DeauthorizeData redshift : Profil DeleteAuthentication décalage vers le rouge : DeleteCluster décalage vers le rouge : DeleteCluster ParameterGroup décalage vers le rouge : DeleteCluster SecurityGroup redshift : DeleteCluster Instantané décalage vers le rouge : DeleteCluster SubnetGroup décalage vers le rouge : DeleteCustom DomainAssociation redshift : Accès DeleteEndpoint redshift : Abonnement DeleteEvent décalage vers le rouge : DeleteHsm ClientCertificate redshift : Configuration DeleteHsm décalage vers le rouge : DeletePartner redshift : Action DeleteScheduled décalage vers le rouge : DeleteSnapshot CopyGrant

Préfixe de service	Actions
	redshift : Calendrier DeleteSnapshot redshift : limite DeleteUsage redshift : Attributs DescribeAccount redshift : Profils DescribeAuthentication décalage vers le rouge : DescribeCluster DbRevisions décalage vers le rouge : DescribeCluster ParameterGroups redshift : Paramètres DescribeCluster décalage vers le rouge : DescribeClusters décalage vers le rouge : DescribeCluster SecurityGroups redshift : DescribeCluster Instantanés décalage vers le rouge : DescribeCluster SubnetGroups redshift : Pistes DescribeCluster redshift : Versions DescribeCluster décalage vers le rouge : DescribeCustom DomainAssociations redshift : Shares DescribeData redshift : Consommateur DescribeData SharesFor redshift : Producteur DescribeData SharesFor décalage vers le rouge : DescribeDefault ClusterParameters redshift : Accès DescribeEndpoint redshift : Autorisation DescribeEndpoint redshift : Catégories DescribeEvent

Préfixe de service	Actions
	décalage vers le rouge : DescribeEvents redshift : Abonnements DescribeEvent décalage vers le rouge : DescribeHsm ClientCertificates redshift : Configurations DescribeHsm redshift : Intégrations DescribeInbound redshift : État DescribeLogging décalage vers le rouge : DescribeNode ConfigurationOptions décalage vers le rouge : DescribeOrderable ClusterOptions décalage vers le rouge : DescribePartners décalage vers le rouge : DescribeRedshift IdcApplications redshift : État DescribeReserved NodeExchange décalage vers le rouge : DescribeReserved NodeOfferings redshift : Nœuds DescribeReserved décalage vers le rouge : DescribeResize redshift : Actions DescribeScheduled décalage vers le rouge : DescribeSnapshot CopyGrants redshift : Horaires DescribeSnapshot décalage vers le rouge : DescribeStorage décalage vers le rouge : DescribeTable RestoreStatus redshift : Limites DescribeUsage décalage vers le rouge : DisableLogging

Préfixe de service	Actions
	redshift : Copier DisableSnapshot décalage vers le rouge : DisassociateData ShareConsumer décalage vers le rouge : EnableLogging redshift : Copier EnableSnapshot redshift : FailoverPrimary Calculer redshift : GetCluster Informations d'identification redshift : IAM GetCluster CredentialsWith décalage vers le rouge : GetReserved NodeExchange Configura tionOptions redshift : Offres GetReserved NodeExchange décalage vers le rouge : ListRecommendations redshift : Configuration ModifyAqua redshift : Profil ModifyAuthentication décalage vers le rouge : ModifyCluster décalage vers le rouge : ModifyCluster DbRevision décalage vers le rouge : ModifyCluster IamRoles redshift : Entretien ModifyCluster décalage vers le rouge : ModifyCluster ParameterGroup redshift : ModifyCluster Instantané décalage vers le rouge : ModifyCluster SnapshotSchedule décalage vers le rouge : ModifyCluster SubnetGroup

Préfixe de service	Actions
	décalage vers le rouge : ModifyCustom DomainAssociation redshift : Accès ModifyEndpoint redshift : Abonnement ModifyEvent redshift : Action ModifyScheduled redshift : Période ModifySnapshot CopyRetention redshift : Calendrier ModifySnapshot redshift : limite ModifyUsage décalage vers le rouge : PauseCluster décalage vers le rouge : PurchaseReserved NodeOffering décalage vers le rouge : RebootCluster redshift : Partager RejectData décalage vers le rouge : ResetCluster ParameterGroup décalage vers le rouge : ResizeCluster décalage vers le rouge : RestoreFrom ClusterSnapshot redshift : RestoreTable FromCluster Instantané décalage vers le rouge : ResumeCluster redshift : Ingress RevokeCluster SecurityGroup redshift : Accès RevokeEndpoint redshift : Accès RevokeSnapshot redshift : clé RotateEncryption redshift : État UpdatePartner

Préfixe de service	Actions
redshift-data	redshift-data : Déclaration BatchExecute
	données redshift : CancelStatement
	données redshift : DescribeStatement
	données redshift : DescribeTable
	données redshift : ExecuteStatement
	redshift-data : Résultat GetStatement
	données redshift : ListDatabases
	données redshift : ListSchemas
	données redshift : ListStatements
	données redshift : ListTables

Préfixe de service	Actions
refactor-spaces	espaces de refactorisation : CreateApplication espaces de refactorisation : CreateEnvironment espaces de refactorisation : CreateRoute espaces de refactorisation : CreateService espaces de refactorisation : DeleteApplication espaces de refactorisation : DeleteEnvironment refactor-spaces : Politique DeleteResource espaces de refactorisation : DeleteRoute espaces de refactorisation : DeleteService espaces de refactorisation : GetApplication espaces de refactorisation : GetEnvironment refactor-spaces : Politique GetResource espaces de refactorisation : GetRoute espaces de refactorisation : GetService espaces de refactorisation : ListApplications espaces de refactorisation : ListEnvironments espaces de refactorisation : Vpcs ListEnvironment espaces de refactorisation : ListRoutes espaces de refactorisation : ListServices refactor-spaces : Politique PutResource espaces de refactorisation : UpdateRoute

Préfixe de service	Actions
rekognition	reconnaissance : AssociateFaces
	reconnaissance : CompareFaces
	reconnaissance : Version CopyProject
	reconnaissance : CreateCollection
	reconnaissance : CreateDataset
	reconnaissance : CreateFace LivenessSession
	reconnaissance : CreateProject
	reconnaissance : Version CreateProject
	Rekognition : Processeur CreateStream
	reconnaissance : CreateUser
	reconnaissance : DeleteCollection
	reconnaissance : DeleteDataset
	reconnaissance : DeleteFaces
	reconnaissance : DeleteProject
	reconnaissance : Politique DeleteProject
	reconnaissance : Version DeleteProject
	Rekognition : Processeur DeleteStream
	reconnaissance : DeleteUser
	reconnaissance : DescribeCollection
	reconnaissance : DescribeDataset
	reconnaissance : DescribeProjects

Préfixe de service	Actions
	reconnaissance : Versions DescribeProject
	Rekognition : Processeur DescribeStream
	Rekognition : Étiquettes DetectCustom
	reconnaissance : DetectFaces
	reconnaissance : DetectLabels
	Rekognition : Étiquettes DetectModeration
	Rekognition : Équipement DetectProtective
	reconnaissance : DetectText
	reconnaissance : DisassociateFaces
	reconnaissance : Entrées DistributeDataset
	reconnaissance : Info GetCelebrity
	reconnaissance : Reconnaissance GetCelebrity
	reconnaissance : Modération GetContent
	reconnaissance : Détection GetFace
	rekognition : Résultats GetFace LivenessSession
	Rekognition : Rechercher GetFace
	reconnaissance : Détection GetLabel
	reconnaissance : GetMedia AnalysisJob
	Rekognition : Suivi GetPerson
	reconnaissance : Détection GetSegment
	reconnaissance : Détection GetText

Préfixe de service	Actions
	reconnaissance : IndexFaces
	reconnaissance : ListCollections
	reconnaissance : Entrées ListDataset
	Rekognition : Étiquettes ListDataset
	reconnaissance : ListFaces
	reconnaissance : ListMedia AnalysisJobs
	Rekognition : Politiques ListProject
	reconnaissance : Processeurs ListStream
	reconnaissance : ListUsers
	reconnaissance : Politique PutProject
	reconnaissance : RecognizeCelebrities
	reconnaissance : SearchFaces
	reconnaissance : SearchFaces ByImage
	reconnaissance : SearchUsers
	reconnaissance : SearchUsers ByImage
	reconnaissance : Reconnaissance StartCelebrity
	reconnaissance : Modération StartContent
	reconnaissance : Détection StartFace
	reconnaissance : StartFace LivenessSession
	Rekognition : Rechercher StartFace
	reconnaissance : Détection StartLabel

Préfixe de service	Actions
	reconnaissance : StartMedia AnalysisJob
	Rekognition : Suivi StartPerson
	reconnaissance : Version StartProject
	reconnaissance : Détection StartSegment
	Rekognition : Processeur StartStream
	reconnaissance : Détection StartText
	reconnaissance : Version StopProject
	Rekognition : Processeur StopStream
	reconnaissance : Entrées UpdateDataset
	Rekognition : Processeur UpdateStream

Préfixe de service	Actions
resiliencehub	hub de résilience : AddDraft AppVersion ResourceMappings hub de résilience : CreateApp resiliencehub : Composant CreateApp VersionApp hub de résilience : CreateApp VersionResource resiliencehub : Modèle CreateRecommendation resiliencehub : Politique CreateResiliency hub de résilience : DeleteApp resiliencehub : Évaluation DeleteApp hub de résilience : DeleteApp InputSource resiliencehub : Composant DeleteApp VersionApp hub de résilience : DeleteApp VersionResource resiliencehub : Modèle DeleteRecommendation resiliencehub : Politique DeleteResiliency hub de résilience : DescribeApp resiliencehub : Évaluation DescribeApp Resiliencehub : Version DescribeApp resiliencehub : Composant DescribeApp VersionApp hub de résilience : DescribeApp VersionResource hub de résilience : DescribeApp VersionResources Resolutio nStatus hub de résilience : DescribeApp VersionTemplate

Préfixe de service	Actions
	resiliencehub : État DescribeDraft AppVersion ResourcesImport resiliencehub : Politique DescribeResiliency hub de résilience : ImportResources ToDraft AppVersion resiliencehub : recommandations ListAlarm resiliencehub : Évaluations ListApp hub de résilience : ListApp ComponentCompliances hub de résilience : ListApp ComponentRecommendations hub de résilience : ListApp InputSources hub de résilience : ListApps resiliencehub : Composants ListApp VersionApp resiliencehub : Mappages ListApp VersionResource hub de résilience : ListApp VersionResources resiliencehub : Versions ListApp resiliencehub : Modèles ListRecommendation resiliencehub : Politiques ListResiliency resiliencehub : recommandations ListSop hub de résilience : ListSuggested ResiliencyPolicies resiliencehub : recommandations ListTest resiliencehub : Ressources ListUnsupported AppVersion Resiliencehub : Version PublishApp resiliencehub : Modèle PutDraft AppVersion

Préfixe de service	Actions
	hub de résilience : RemoveDraft AppVersion ResourceMappings hub de résilience : ResolveApp VersionResources resiliencehub : Évaluation StartApp hub de résilience : UpdateApp Resiliencehub : Version UpdateApp resiliencehub : Composant UpdateApp VersionApp hub de résilience : UpdateApp VersionResource resiliencehub : Politique UpdateResiliency

Préfixe de service	Actions
resource-explorer-2	resource-explorer-2 : Afficher AssociateDefault
	resource-explorer-2 : Afficher BatchGet
	explorateur de ressources-2 : CreateIndex
	explorateur de ressources-2 : CreateView
	explorateur de ressources-2 : DeleteIndex
	explorateur de ressources-2 : DeleteView
	resource-explorer-2 : Afficher DisassociateDefault
	explorateur de ressources 2 : Configuration GetAccount LevelServ ice
	resource-explorer-2 : Afficher GetDefault
	explorateur de ressources-2 : GetIndex
	explorateur de ressources-2 : ListIndexes
	explorateur de ressources-2 : ListIndexes ForMembers
	explorateur de ressources-2 : ListSupported ResourceTypes
	explorateur de ressources-2 : ListViews
	resource-explorer-2:Search
	explorateur de ressources 2 : tapez UpdateIndex
	explorateur de ressources-2 : UpdateView

Préfixe de service	Actions
resource-groups	groupes de ressources : CreateGroup groupes de ressources : DeleteGroup groupes de ressources : Paramètres GetAccount groupes de ressources : GetGroup groupes de ressources : Configuration GetGroup resource-groups : Requête GetGroup groupes de ressources : GroupResources groupes de ressources : Ressources ListGroup groupes de ressources : ListGroups groupes de ressources : Configuration PutGroup groupes de ressources : SearchResources groupes de ressources : UngroupResources groupes de ressources : Paramètres UpdateAccount groupes de ressources : UpdateGroup resource-groups : Requête UpdateGroup

Préfixe de service	Actions
robomaker	Robomaker : Worlds BatchDelete
	fabricant de robots : BatchDescribe SimulationJob
	robomaker : Job CancelDeployment
	robomaker : Job CancelSimulation
	fabricant de robots : CancelSimulation JobBatch
	fabricant de robots : CancelWorld ExportJob
	fabricant de robots : CancelWorld GenerationJob
	robomaker : Job CreateDeployment
	fabricant de robots : CreateFleet
	fabricant de robots : CreateRobot
	Robomaker : Application CreateRobot
	fabricant de robots : CreateRobot ApplicationVersion
	Robomaker : Application CreateSimulation
	fabricant de robots : CreateSimulation ApplicationVersion
	robomaker : Job CreateSimulation
	fabricant de robots : CreateWorld ExportJob
	fabricant de robots : CreateWorld GenerationJob
	robomaker : Modèle CreateWorld
	fabricant de robots : DeleteFleet
	fabricant de robots : DeleteRobot
	Robomaker : Application DeleteRobot

Préfixe de service	Actions
	Robomaker : Application DeleteSimulation
	robomaker : Modèle DeleteWorld
	fabricant de robots : DeregisterRobot
	robomaker : Job DescribeDeployment
	fabricant de robots : DescribeFleet
	fabricant de robots : DescribeRobot
	Robomaker : Application DescribeRobot
	Robomaker : Application DescribeSimulation
	robomaker : Job DescribeSimulation
	fabricant de robots : DescribeSimulation JobBatch
	fabricant de robots : DescribeWorld
	fabricant de robots : DescribeWorld ExportJob
	fabricant de robots : DescribeWorld GenerationJob
	robomaker : Modèle DescribeWorld
	fabricant de robots : GetWorld TemplateBody
	robomaker : Offres d'emploi ListDeployment
	fabricant de robots : ListFleets
	robomaker : Applications ListRobot
	fabricant de robots : ListRobots
	robomaker : Applications ListSimulation
	fabricant de robots : ListSimulation JobBatches

Préfixe de service	Actions
	robomaker : Offres d'emploi ListSimulation
	fabricant de robots : ListWorld ExportJobs
	fabricant de robots : ListWorld GenerationJobs
	fabricant de robots : ListWorlds
	robomaker : Modèles ListWorld
	fabricant de robots : RegisterRobot
	robomaker : Job RestartSimulation
	fabricant de robots : StartSimulation JobBatch
	robomaker : Job SyncDeployment
	Robomaker : Application UpdateRobot
	Robomaker : Application UpdateSimulation
	robomaker : Modèle UpdateWorld

Préfixe de service	Actions
rolesanywhere	rôles n'importe où : CreateProfile rolesanywhere : Anchor CreateTrust rolesanywhere : Cartographie DeleteAttribute rôles n'importe où : DeleteCrl rôles n'importe où : DeleteProfile rolesanywhere : Anchor DeleteTrust rôles n'importe où : DisableCrl rôles n'importe où : DisableProfile rolesanywhere : Anchor DisableTrust rôles n'importe où : EnableCrl rôles n'importe où : EnableProfile rolesanywhere : Anchor EnableTrust rôles n'importe où : GetCrl rôles n'importe où : GetProfile rôles n'importe où : GetSubject rolesanywhere : Anchor GetTrust rôles n'importe où : ImportCrl rôles n'importe où : ListCrls rôles n'importe où : ListProfiles rôles n'importe où : ListSubjects rolesanywhere : Anchors ListTrust

Préfixe de service	Actions
	rolesanywhere : Cartographie PutAttribute rolesanywhere : Paramètres PutNotification rolesanywhere : Paramètres ResetNotification rôles n'importe où : UpdateCrl rôles n'importe où : UpdateProfile rolesanywhere : Anchor UpdateTrust

Préfixe de service	Actions
route53	route 53 : ActivateKey SigningKey Route 53 : zone VPC associée WithHosted route 53 : Collection ChangeCidr route 53 : ChangeResource RecordSets route 53 : Collection CreateCidr route53 : Vérifier CreateHealth route 53 : Zone CreateHosted route 53 : CreateKey SigningKey route 53 : CreateQuery LoggingConfig route 53 : CreateReusable DelegationSet route53 : Politique CreateTraffic route 53 : CreateTraffic PolicyInstance route 53 : CreateTraffic PolicyVersion Route 53 : créer un VPC AssociationAuthorization route 53 : DeactivateKey SigningKey route 53 : Collection DeleteCidr route53 : Vérifier DeleteHealth route 53 : Zone DeleteHosted route 53 : DeleteKey SigningKey route 53 : DeleteQuery LoggingConfig route 53 : DeleteReusable DelegationSet

Préfixe de service	Actions
	route53 : Politique DeleteTraffic
	route 53 : DeleteTraffic PolicyInstance
	Route 53 : Supprimer le VPC AssociationAuthorization
	route 53 : Zone DNSSEC DisableHosted
	Route 53 FromHosted : dissocier la zone VPC
	route 53 : Zone DNSSEC EnableHosted
	route 53 : Limite GetAccount
	route 53 : GetChange
	route 53 : GetChecker IpRanges
	route53:GetDNSSEC
	route 53 : Localisation GetGeo
	route53 : Vérifier GetHealth
	route 53 : GetHealth CheckCount
	route 53 : GetHealth CheckLast FailureReason
	route 53 : GetHealth CheckStatus
	route 53 : Zone GetHosted
	route 53 : GetHosted ZoneCount
	route 53 : GetHosted ZoneLimit
	route 53 : GetQuery LoggingConfig
	route 53 : GetReusable DelegationSet
	route 53 : Limite GetReusable DelegationSet

Préfixe de service	Actions
	route53 : Politique GetTraffic
	route 53 : GetTraffic PolicyInstance
	route 53 : Compter GetTraffic PolicyInstance
	route 53 : Blocs ListCidr
	route 53 : Collectes ListCidr
	route53 : Localisations ListCidr
	route53 : Localisations ListGeo
	route53 : Chèques ListHealth
	route 53 : Zones ListHosted
	route53 : Nom ListHosted ZonesBy
	route 53 : VPC ListHosted ZonesBy
	route 53 : ListQuery LoggingConfigs
	route 53 : ListResource RecordSets
	route 53 : ListReusable DelegationSets
	route53 : Politiques ListTraffic
	route 53 : ListTraffic PolicyInstances
	route 53 : Zone ListTraffic PolicyInstances ByHosted
	route 53 : ListTraffic PolicyInstances ByPolicy
	route 53 : ListTraffic PolicyVersions
	Route 53 : ListeVPC AssociationAuthorizations
	route53:TestDNSAnswer

Préfixe de service	Actions
	route53 : Vérifier UpdateHealth route 53 : UpdateHosted ZoneComment route 53 : UpdateTraffic PolicyComment route 53 : UpdateTraffic PolicyInstance

Préfixe de service	Actions
route53-recovery-control-config	route53-recovery-control-config : CreateCluster route53-recovery-control-config : Panneau CreateControl route53-recovery-control-config : Contrôle CreateRouting route53-recovery-control-config : Règle CreateSafety route53-recovery-control-config : DeleteCluster route53-recovery-control-config : Panneau DeleteControl route53-recovery-control-config : Contrôle DeleteRouting route53-recovery-control-config : Règle DeleteSafety route53-recovery-control-config : DescribeCluster route53-recovery-control-config : Panneau DescribeControl route53-recovery-control-config : Contrôle DescribeRouting route53-recovery-control-config : Règle DescribeSafety route53-recovery-control-config : Politique GetResource route53-recovery-control-config : Route53 ListAssociated HealthChecks route53-recovery-control-config : ListClusters route53-recovery-control-config : Panneaux ListControl route53-recovery-control-config : Contrôles ListRouting route53-recovery-control-config : Règles ListSafety route53-recovery-control-config : Panneau UpdateControl route53-recovery-control-config : Contrôle UpdateRouting

Préfixe de service	Actions
	route53-recovery-control-config : Règle UpdateSafety

Préfixe de service	Actions
route53-recovery-readiness	route 53 - préparation au rétablissement : CreateCell route 53 - préparation au rétablissement : CreateCross AccountAuthorization route53 - préparation à la restauration : vérifier CreateReadiness route 53 - préparation au rétablissement : Groupe CreateRecovery route53 - état de préparation à la restauration : défini CreateResource route 53 - préparation au rétablissement : DeleteCell route 53 - préparation au rétablissement : DeleteCross AccountAuthorization route53 - préparation à la restauration : vérifier DeleteReadiness route 53 - préparation au rétablissement : Groupe DeleteRecovery route53 - état de préparation à la restauration : défini DeleteResource route 53 - préparation au rétablissement : recommandations GetArchitecture route 53 - préparation au rétablissement : GetCell route 53 - préparation au rétablissement : GetCell Readiness Summary route53 - préparation à la restauration : vérifier GetReadiness route53-recoory-readiness : État GetReadiness CheckResource route 53 - préparation au rétablissement : GetReadiness CheckStatus route 53 - préparation au rétablissement : Groupe GetRecovery

Préfixe de service	Actions
	route53-Recovery-Readiness : Résumé GetRecovery GroupReadiness route53 - état de préparation à la restauration : défini GetResource route 53 - préparation au rétablissement : ListCells route 53 - préparation au rétablissement : ListCross AccountAuthorizations route53 - préparation à la restauration : vérifications ListReadiness route 53 - préparation à la restauration : groupes ListRecovery route53 - préparation à la restauration : ensembles ListResource route 53 - préparation au rétablissement : ListRules route 53 - préparation au rétablissement : UpdateCell route53 - préparation à la restauration : vérifier UpdateReadiness route 53 - préparation au rétablissement : Groupe UpdateRecovery route53 - état de préparation à la restauration : défini UpdateResource

Préfixe de service	Actions
route53resolver	résolveur Route53 : AssociateFirewall RuleGroup route53resolver : Adresse AssociateResolver EndpointIp route53resolver : Config AssociateResolver QueryLog route53resolver : Règle AssociateResolver résolveur Route53 : CreateFirewall DomainList route53resolver : Règle CreateFirewall résolveur Route53 : CreateFirewall RuleGroup route53resolver : point de terminaison CreateResolver route53resolver : Config CreateResolver QueryLog route53resolver : Règle CreateResolver résolveur Route53 : DeleteFirewall DomainList route53resolver : Règle DeleteFirewall résolveur Route53 : DeleteFirewall RuleGroup route53resolver : résolveur DeleteOutpost route53resolver : point de terminaison DeleteResolver route53resolver : Config DeleteResolver QueryLog route53resolver : Règle DeleteResolver résolveur Route53 : DisassociateFirewall RuleGroup route53resolver : Adresse DisassociateResolver EndpointIp route53resolver : Config DisassociateResolver QueryLog route53resolver : Règle DisassociateResolver

Préfixe de service	Actions
	route53resolver : Config GetFirewall résolveur Route53 : GetFirewall DomainList résolveur Route53 : GetFirewall RuleGroup résolveur route53 : Association GetFirewall RuleGroup route53resolver : Politique GetFirewall RuleGroup route53resolver : résolveur GetOutpost route53resolver : Config GetResolver résolveur Route53 : GetResolver DnssecConfig route53resolver : point de terminaison GetResolver route53resolver : Config GetResolver QueryLog résolveur Route53 : GetResolver QueryLog ConfigAssociation résolveur Route53 : GetResolver QueryLog ConfigPolicy route53resolver : Règle GetResolver résolveur Route53 : GetResolver RuleAssociation résolveur Route53 : GetResolver RulePolicy route53resolver : Domaines ImportFirewall route53resolver : Configurations ListFirewall résolveur Route53 : ListFirewall DomainLists route53resolver : Domaines ListFirewall résolveur route53 : Associations ListFirewall RuleGroup résolveur Route53 : ListFirewall RuleGroups

Préfixe de service	Actions
	route53resolver : Règles ListFirewall route53resolver : résolveurs ListOutpost route53resolver : Configurations ListResolver résolveur Route53 : ListResolver DnssecConfigs route53resolver : Adresses ListResolver EndpointIp route53resolver : points de terminaison ListResolver résolveur Route53 : ListResolver QueryLog ConfigAssociations route53resolver : Configurations ListResolver QueryLog résolveur Route53 : ListResolver RuleAssociations route53resolver : Règles ListResolver route53resolver : Politique PutFirewall RuleGroup résolveur Route53 : PutResolver QueryLog ConfigPolicy route53resolver : Config UpdateFirewall route53resolver : Domaines UpdateFirewall route53resolver : Règle UpdateFirewall résolveur route53 : Association UpdateFirewall RuleGroup route53resolver : résolveur UpdateOutpost route53resolver : Config UpdateResolver résolveur Route53 : UpdateResolver DnssecConfig route53resolver : point de terminaison UpdateResolver route53resolver : Règle UpdateResolver

Préfixe de service	Actions
rum	rhum : BatchCreate RumMetric Définitions rhum : BatchDelete RumMetric Définitions rhum : BatchGet RumMetric Définitions Rhum : CreateApp Moniteur Rhum : DeleteApp Moniteur rhum : DeleteRum MetricsDestination Rhum : GetApp Moniteur rhum : GetApp MonitorData rhum : ListApp Moniteurs rhum : ListRum MetricsDestinations rhum : PutRum MetricsDestination Rhum : UpdateApp Moniteur rhum : UpdateRum MetricDefinition

Préfixe de service	Actions
s3	s3 : AssociateAccess GrantsIdentity Centre s3 : CreateAccess Subvention s3 : CreateAccess GrantsInstance s3 : CreateAccess GrantsLocation s3 : CreateAccess Point s3 : CreateAccess PointFor ObjectLambda s3 : CreateBucket s3 : CreateJob s3 : CreateMulti RegionAccess Point s3 : DeleteAccess Subvention s3 : DeleteAccess GrantsInstance s3 : DeleteAccess GrantsInstance ResourcePolicy s3 : DeleteAccess GrantsLocation s3 : DeleteAccess Point s3 : DeleteAccess PointFor ObjectLambda s3 : DeleteAccess PointPolicy s3 : DeleteAccess PointPolicy ForObject Lambda s3 : PutAccount PublicAccess Bloquer s3 : DeleteBucket s3 : PutAnalytics Configuration s3 : PutBucket CORS

Préfixe de service	Actions
	s3 : PutEncryption Configuration s3 : PutIntelligent TieringConfiguration s3 : PutInventory Configuration s3 : PutLifecycle Configuration s3 : PutMetrics Configuration s3 : PutBucket OwnershipControls s3 : DeleteBucket Politique s3 : PutBucket PublicAccess Bloquer s3 : PutReplication Configuration s3 : DeleteBucket Site Web s3 : DeleteMulti RegionAccess Point s3 : DeleteStorage LensConfiguration s3 : DescribeJob s3 : DescribeMulti RegionAccess PointOperation s3 : DissociateAccess GrantsIdentity Centre s3 : GetAccelerate Configuration s3 : GetAccess Subvention s3 : GetAccess GrantsInstance s3 : GetAccess GrantsInstance ForPrefix s3 : GetAccess GrantsInstance ResourcePolicy s3 : GetAccess GrantsLocation

Préfixe de service	Actions
	s3 : GetAccess Point s3 : GetAccess PointConfiguration ForObject Lambda s3 : GetAccess PointFor ObjectLambda s3 : GetAccess PointPolicy s3 : GetAccess PointPolicy ForObject Lambda s3 : GetAccess PointPolicy État s3 : GetAccess PointPolicy StatusFor ObjectLambda s3 : GetAccount PublicAccess Bloquer s3 : GetBucket Acl s3 : GetAnalytics Configuration s3 : GetBucket CORS s3 : GetEncryption Configuration s3 : GetIntelligent TieringConfiguration s3 : GetInventory Configuration s3 : GetLifecycle Configuration s3 : GetBucket Emplacement s3 : GetBucket Journalisation s3 : GetMetrics Configuration s3 : GetBucket Notification s3 : GetBucket ObjectLock Configuration s3 : GetBucket OwnershipControls

Préfixe de service	Actions
	s3 : GetBucket Politique s3 : GetBucket PolicyStatus s3 : GetBucket PublicAccess Bloquer s3 : GetReplication Configuration s3 : GetBucket RequestPayment s3 : GetBucket Gestion des versions s3 : GetBucket Site Web s3 : GetData Accès s3 : GetMulti RegionAccess Point s3 : GetMulti RegionAccess PointPolicy s3 : GetMulti RegionAccess PointPolicy État s3 : GetMulti RegionAccess PointRoutes s3 : GetObject Attributs s3 : GetStorage LensConfiguration s3 : GetStorage LensDashboard s3 : ListAccess Subventions s3 : ListAccess GrantsInstances s3 : ListAccess GrantsLocations s3 : ListAccess Points s3 : ListAccess PointsFor ObjectLambda s3 : ListAll MyBuckets

Préfixe de service	Actions
	s3 : ListJobs s3 : ListBucket MultipartUploads s3 : ListMulti RegionAccess Points s3 : ListStorage LensConfigurations s3 : PutAccelerate Configuration s3 : PutAccess GrantsInstance ResourcePolicy s3 : PutAccess PointConfiguration ForObject Lambda s3 : PutAccess PointPolicy s3 : PutAccess PointPolicy ForObject Lambda s3 : PutAccount PublicAccess Bloquer s3 : PutBucket Acl s3 : PutAnalytics Configuration s3 : PutBucket CORS s3 : PutEncryption Configuration s3 : PutIntelligent TieringConfiguration s3 : PutInventory Configuration s3 : PutLifecycle Configuration s3 : PutBucket Journalisation s3 : PutMetrics Configuration s3 : PutBucket Notification s3 : PutBucket ObjectLock Configuration

Préfixe de service	Actions
	s3 : PutBucket OwnershipControls s3 : PutBucket Politique s3 : PutBucket PublicAccess Bloquer s3 : PutReplication Configuration s3 : PutBucket RequestPayment s3 : PutBucket Gestion des versions s3 : PutBucket Site Web s3 : PutMulti RegionAccess PointPolicy s3 : PutStorage LensConfiguration s3 : SubmitMulti RegionAccess PointRoutes s3 : UpdateAccess GrantsLocation s3 : UpdateJob Priorité s3 : UpdateJob État
s3-outposts	avant-postes S3 : CreateEndpoint avant-postes S3 : DeleteEndpoint avant-postes S3 : ListEndpoints avant-postes S3 : avec S3 ListOutposts s3-outposts : points de terminaison ListShared

Préfixe de service	Actions
sagemaker-geospatial	sagemaker-geospatial : DeleteEarth ObservationJob sagemaker-geospatial : DeleteVector EnrichmentJob sagemaker-geospatial : ExportEarth ObservationJob sagemaker-geospatial : ExportVector EnrichmentJob sagemaker-geospatial : GetEarth ObservationJob sagemaker-geospatial : GetRaster DataCollection sagemaker-geospatial : GetTile sagemaker-geospatial : GetVector EnrichmentJob sagemaker-geospatial : ListEarth ObservationJobs sagemaker-geospatial : ListRaster DataCollections sagemaker-geospatial : ListVector EnrichmentJobs sagemaker-geospatial : SearchRaster DataCollection sagemaker-geospatial : StartEarth ObservationJob sagemaker-geospatial : StartVector EnrichmentJob sagemaker-geospatial : StopEarth ObservationJob sagemaker-geospatial : StopVector EnrichmentJob

Préfixe de service	Actions
savingsplans	plans d'épargne : Plan CreateSavings
	plans d'épargne : DeleteQueued SavingsPlan
	plans d'épargne : DescribeSavings PlanRates
	plans d'épargne : Plans DescribeSavings
	plans d'épargne : Tarifs DescribeSavings PlansOffering
	plans d'épargne : DescribeSavings PlansOfferings
	plans d'épargne : Plan ReturnSavings

Préfixe de service	Actions
schemas	schémas : CreateDiscoverer schémas : CreateRegistry schémas : CreateSchema schémas : DeleteDiscoverer schémas : DeleteRegistry schémas : Politique DeleteResource schémas : DeleteSchema schémas : Version DeleteSchema schémas : Binding DescribeCode schémas : DescribeDiscoverer schémas : DescribeRegistry schémas : DescribeSchema schémas : ExportSchema schémas : GetCode BindingSource schémas : GetDiscovered Schéma schémas : Politique GetResource schémas : ListDiscoverers schémas : ListRegistries schémas : ListSchemas schémas : Versions ListSchema schémas : Binding PutCode

Préfixe de service	Actions
	schémas : Politique PutResource schémas : SearchSchemas schémas : StartDiscoverer schémas : StopDiscoverer schémas : UpdateDiscoverer schémas : UpdateRegistry schémas : UpdateSchema
sdb	sdb : CreateDomain sdb : DeleteDomain sdb : DomainMetadata sdb : ListDomains

Préfixe de service	Actions
secretsmanager	secretsmanager : Secret CancelRotate responsable des secrets : CreateSecret secretsmanager : Politique DeleteResource responsable des secrets : DeleteSecret responsable des secrets : DescribeSecret secretsmanager : Mot de passe GetRandom secretsmanager : Politique GetResource secretsmanager : Valeur GetSecret responsable des secrets : ListSecrets responsable des secrets : ListSecret VersionIds secretsmanager : Politique PutResource secretsmanager : Valeur PutSecret responsable des secrets : RemoveRegions FromReplication responsable des secrets : ReplicateSecret ToRegions responsable des secrets : RestoreSecret responsable des secrets : RotateSecret responsable des secrets : StopReplication ToReplica responsable des secrets : UpdateSecret secretsmanager : Politique ValidateResource

Préfixe de service	Actions
securityhub	securityhub : Invitation AcceptAdministrator
	hub de sécurité : AcceptInvitation
	hub de sécurité : BatchDelete AutomationRules
	securityhub : Normes BatchDisable
	securityhub : Normes BatchEnable
	hub de sécurité : BatchGet AutomationRules
	securityhub : Associations BatchGet ConfigurationPolicy
	hub de sécurité : BatchGet SecurityControls
	securityhub : Associations BatchGet StandardsControl
	securityhub : Conclusions BatchImport
	hub de sécurité : BatchUpdate AutomationRules
	securityhub : Conclusions BatchUpdate
	securityhub : Associations BatchUpdate StandardsControl
	securityhub : Cible CreateAction
	securityhub : CreateAutomation Règle
	securityhub : Politique CreateConfiguration
	securityhub : CreateFinding agrégateur
	hub de sécurité : CreateInsight
	hub de sécurité : CreateMembers
	hub de sécurité : DeclineInvitations
	securityhub : Cible DeleteAction

Préfixe de service	Actions
	securityhub : Politique DeleteConfiguration
	securityhub : DeleteFinding agrégateur
	hub de sécurité : DeleteInsight
	hub de sécurité : DeleteInvitations
	hub de sécurité : DeleteMembers
	securityhub : Cibles DescribeAction
	hub de sécurité : DescribeHub
	securityhub : Configuration DescribeOrganization
	hub de sécurité : DescribeProducts
	hub de sécurité : DescribeStandards
	securityhub : Produit DisableImport FindingsFor
	hub de sécurité : DisableOrganization AdminAccount
	hub de sécurité : DisableSecurity Hub
	hub de sécurité : DisassociateFrom AdministratorAccount
	hub de sécurité : DisassociateFrom MasterAccount
	hub de sécurité : DisassociateMembers
	securityhub : Produit EnableImport FindingsFor
	hub de sécurité : EnableOrganization AdminAccount
	hub de sécurité : EnableSecurity Hub
	securityhub : Compte GetAdministrator
	securityhub : Politique GetConfiguration

Préfixe de service	Actions
	hub de sécurité : GetConfiguration PolicyAssociation
	securityhub : Normes GetEnabled
	securityhub : GetFinding agrégateur
	securityhub : Historique GetFinding
	hub de sécurité : GetFindings
	securityhub : Résultats GetInsight
	hub de sécurité : GetInsights
	securityhub : Nombre GetInvitations
	securityhub : Compte GetMaster
	hub de sécurité : GetMembers
	hub de sécurité : GetSecurity ControlDefinition
	hub de sécurité : InviteMembers
	securityhub : Règles ListAutomation
	securityhub : Politiques ListConfiguration
	hub de sécurité : ListConfiguration PolicyAssociations
	securityhub : Importer ListEnabled ProductsFor
	securityhub : Agrégateurs ListFinding
	hub de sécurité : ListInvitations
	hub de sécurité : ListMembers
	hub de sécurité : ListOrganization AdminAccounts
	hub de sécurité : ListSecurity ControlDefinitions

Préfixe de service	Actions
	hub de sécurité : ListStandards ControlAssociations
	hub de sécurité : StartConfiguration PolicyAssociation
	hub de sécurité : StartConfiguration PolicyDisassociation
	securityhub : Cible UpdateAction
	securityhub : Politique UpdateConfiguration
	securityhub : UpdateFinding agrégateur
	hub de sécurité : UpdateFindings
	hub de sécurité : UpdateInsight
	securityhub : Configuration UpdateOrganization
	securityhub : Contrôle UpdateSecurity
	hub de sécurité : UpdateSecurity HubConfiguration

Préfixe de service	Actions
securitylake	lac de sécurité : CreateAws LogSource lac de sécurité : CreateCustom LogSource securitylake : Abonnement CreateData LakeException securitylake : Configuration CreateData LakeOrganization lac de sécurité : CreateSubscriber securitylake : Notification CreateSubscriber lac de sécurité : DeleteAws LogSource lac de sécurité : DeleteCustom LogSource securitylake : Abonnement DeleteData LakeException securitylake : Configuration DeleteData LakeOrganization lac de sécurité : DeleteSubscriber securitylake : Notification DeleteSubscriber securitylake : Administrateur DeregisterData LakeDelegated securitylake : Abonnement GetData LakeException securitylake : Configuration GetData LakeOrganization lac de sécurité : GetData LakeSources lac de sécurité : GetSubscriber lac de sécurité : lacs ListData securitylake : Sources ListLog lac de sécurité : ListSubscribers securitylake : Administrateur RegisterData LakeDelegated

Préfixe de service	Actions
	securitylake : Abonnement UpdateData LakeException lac de sécurité : UpdateSubscriber securitylake : Notification UpdateSubscriber
serverlessrepo	dépôt sans serveur : CreateApplication serverlessrepo : Version CreateApplication serverlessrepo : Set CreateCloud FormationChange dépôt sans serveur : CreateCloud FormationTemplate dépôt sans serveur : DeleteApplication dépôt sans serveur : GetApplication serverlessrepo : Politique GetApplication dépôt sans serveur : GetCloud FormationTemplate serverlessrepo : Dépendances ListApplication dépôt sans serveur : ListApplications serverlessrepo : Versions ListApplication serverlessrepo : Politique PutApplication dépôt sans serveur : UnshareApplication dépôt sans serveur : UpdateApplication

Préfixe de service	Actions
servicecatalog	catalogue de services : Partager AcceptPortfolio catalogue de services : AssociateBudget WithResource catalogue de services : AssociatePrincipal WithPortfolio catalogue de services : AssociateProduct WithPortfolio catalogue de services : AssociateService ActionWith Provision ingArtifact catalogue de services : Artifact BatchAssociate ServiceAction WithProvisioning catalogue de services : Artifact BatchDisassociate ServiceAction FromProvisioning catalogue de services : CopyProduct catalogue de services : CreateConstraint catalogue de services : CreatePortfolio catalogue de services : Partager CreatePortfolio catalogue de services : CreateProduct catalogue de services : CreateProvisioned ProductPlan catalogue de services : Artifact CreateProvisioning catalogue de services : Action CreateService catalogue de services : DeleteConstraint catalogue de services : DeletePortfolio catalogue de services : Partager DeletePortfolio catalogue de services : DeleteProduct

Préfixe de service	Actions
	catalogue de services : DeleteProvisioned ProductPlan catalogue de services : Artifact DeleteProvisioning catalogue de services : Action DeleteService catalogue de services : DescribeConstraint catalogue de services : DescribeCopy ProductStatus catalogue de services : DescribePortfolio catalogue de services : Shares DescribePortfolio catalogue de services : DescribePortfolio ShareStatus catalogue de services : DescribeProduct catalogue de services : DescribeProduct AsAdmin catalogue de services : Afficher DescribeProduct catalogue de services : DescribeProvisioned ProductPlan catalogue de services : Artifact DescribeProvisioning servicecatalog : Paramètres DescribeProvisioning catalogue de services : DescribeRecord catalogue de services : Action DescribeService servicecatalog : Paramètres DescribeService ActionExecution Catalogue de services : Désactiver AWSOrganizationsAccess catalogue de services : DisassociateBudget FromResource catalogue de services : DisassociatePrincipal FromPortfolio catalogue de services : DisassociateProduct FromPortfolio

Préfixe de service	Actions
	catalogue de services : DisassociateService ActionFrom Provision ingArtifact Catalogue de services : activer AWSOrganizationsAccess catalogue de services : ExecuteProvisioned ProductPlan catalogue de services : Action ExecuteProvisioned ProductService Catalogue de services : GET AWSOrganizationsAccessStatus catalogue de services : GetProvisioned ProductOutputs catalogue de services : ImportAs ProvisionedProduct catalogue de services : ListAccepted PortfolioShares catalogue de services : ListBudgets ForResource catalogue de services : ListConstraints ForPortfolio servicecatalog : Chemins ListLaunch catalogue de services : ListOrganization PortfolioAccess catalogue de services : Accès ListPortfolio catalogue de services : ListPortfolios catalogue de services : ListPortfolios ForProduct catalogue de services : ListPrincipals ForPortfolio catalogue de services : ListProvisioned ProductPlans servicecatalog : Artefacts ListProvisioning catalogue de services : ListProvisioning ArtifactsFor ServiceAction servicecatalog : Historique ListRecord

Préfixe de service	Actions
	catalogue de services : Actions ListService catalogue de services : ListService ActionsFor ProvisioningArtifact catalogue de services : ListStack InstancesFor ProvisionedProduct catalogue de services : NotifyProvision ProductEngine WorkflowResult servicecatalog : Résultat NotifyTerminate ProvisionedProduct EngineWorkflow servicecatalog : Résultat NotifyUpdate ProvisionedProduct EngineWorkflow catalogue de services : ProvisionProduct catalogue de services : Partager RejectPortfolio catalogue de services : Produits ScanProvisioned catalogue de services : SearchProducts catalogue de services : SearchProducts AsAdmin catalogue de services : Produits SearchProvisioned catalogue de services : Produit TerminateProvisioned catalogue de services : UpdateConstraint catalogue de services : UpdatePortfolio catalogue de services : Partager UpdatePortfolio catalogue de services : UpdateProduct catalogue de services : Produit UpdateProvisioned catalogue de services : UpdateProvisioned ProductProperties

Préfixe de service	Actions
	catalogue de services : Artifact UpdateProvisioning catalogue de services : Action UpdateService

Préfixe de service	Actions
servicediscovery	servicediscovery : CreateHttp espace de noms
	Découverte des services : CreatePrivate DnsNamespace
	Découverte des services : CreatePublic DnsNamespace
	découverte des services : CreateService
	découverte des services : DeleteNamespace
	découverte des services : DeleteService
	découverte des services : DeregisterInstance
	découverte des services : GetInstance
	Découverte des services : GetInstances HealthStatus
	découverte des services : GetNamespace
	découverte des services : GetOperation
	découverte des services : GetService
	découverte des services : ListInstances
	découverte des services : ListNamespaces
	découverte des services : ListOperations
	découverte des services : ListServices
	découverte des services : RegisterInstance
	servicediscovery : UpdateHttp espace de noms
	servicediscovery : État UpdateInstance CustomHealth
	Découverte des services : UpdatePrivate DnsNamespace
	Découverte des services : UpdatePublic DnsNamespace

Préfixe de service	Actions
	découverte des services : UpdateService
servicequotas	devis de service : AssociateService QuotaTemplate servicequotas : Modèle DeleteService QuotaIncrease RequestFrom devis de service : DisassociateService QuotaTemplate devis de service : GetAssociation ForService QuotaTemplate Quotas de service : GET AWSDefaultServiceQuota servicequotas : Modifier GetRequested ServiceQuota servicequotas : Quota GetService servicequotas : Modèle GetService QuotaIncrease RequestFrom Quotas de service : liste AWSDefaultServiceQuotas devis de service : ListRequested ServiceQuota ChangeHistory devis de service : ListRequested ServiceQuota ChangeHistory ByQuota servicequotas : Modèle ListService QuotaIncrease RequestsIn servicequotas : Quotas ListService devis de service : ListServices servicequotas : Modèle PutService QuotaIncrease RequestInto devis de service : RequestService QuotaIncrease

Préfixe de service	Actions
ses	Utilise : BatchGet MetricData Utilise : CloneReceipt RuleSet Usage : CreateConfiguration Set Usages : CreateConfiguration SetEvent Destination Utilise : CreateConfiguration SetTracking Options Utilise : CreateContact Uses : CreateContact Liste Utilisation : CreateCustom VerificationEmail Modèle Utilise : CreateDedicated IpPool Utilise : CreateDeliverability TestReport Utilisations : CreateEmail Identité Utilise : CreateEmail IdentityPolicy Utilisation : CreateEmail Modèle Utilise : CreateImport Job Utilisation : CreateReceipt Filtre Utilise : CreateReceipt Règle Utilise : CreateReceipt RuleSet Utilise : CreateTemplate Usage : DeleteConfiguration Set Usages : DeleteConfiguration SetEvent Destination Utilise : DeleteConfiguration SetTracking Options

Préfixe de service	Actions
	Utilise : DeleteContact Uses : DeleteContact Liste Utilisation : DeleteCustom VerificationEmail Modèle Utilise : DeleteDedicated IpPool Utilisations : DeleteEmail Identité Utilise : DeleteEmail IdentityPolicy Utilisation : DeleteEmail Modèle Utilise : DeleteIdentity Utilisation : DeleteIdentity Politique Utilisation : DeleteReceipt Filtre Utilise : DeleteReceipt Règle Utilise : DeleteReceipt RuleSet Utilise : DeleteSuppressed Destination Utilise : DeleteTemplate Utilise : DeleteVerified EmailAddress Usages : DescribeActive ReceiptRule Set Usage : DescribeConfiguration Set Utilise : DescribeReceipt Règle Utilise : DescribeReceipt RuleSet Utilise : GetAccount Utilise : GetAccount SendingEnabled

Préfixe de service	Actions
	Utilisations : GetBlacklist Rapports
	Usage : GetConfiguration Set
	Voir : GetConfiguration SetEvent Destinations
	Utilise : GetContact
	Uses : GetContact Liste
	Utilisation : GetCustom VerificationEmail Modèle
	Utilise : GetDedicated Ip
	Utilise : GetDedicated IpPool
	Utilise : GetDedicated Ips
	Utilise : GetDeliverability DashboardOptions
	Utilise : GetDeliverability TestReport
	Utilise : GetDomain DeliverabilityCampaign
	Utilise : GetDomain StatisticsReport
	Utilisations : GetEmail Identité
	Utilise : GetEmail IdentityPolicies
	Utilisation : GetEmail Modèle
	Utilise : GetIdentity DkimAttributes
	Utilise : GetIdentity MailFrom DomainAttributes
	Utilise : GetIdentity NotificationAttributes
	Utilisation : GetIdentity Politiques
	Utilise : GetIdentity VerificationAttributes

Préfixe de service	Actions
	Utilise : GetImport Job
	Utilisations : GetMessage Insights
	Utilise : GetSend Quota
	ses : GetSend Statistiques
	Utilise : GetSuppressed Destination
	Utilise : GetTemplate
	Utilisations : ListConfiguration Sets
	Utilisations : ListContact Listes
	Utilise : ListContacts
	Utilisation : ListCustom VerificationEmail Modèles
	Utilise : ListDedicated IpPools
	Utilise : ListDeliverability TestReports
	Utilise : ListDomain DeliverabilityCampaigns
	Utilisations : ListEmail Identités
	Utilisation : ListEmail Modèles
	Uses : ListExport Offres d'emploi
	Utilise : ListIdentities
	Utilisation : ListIdentity Politiques
	Uses : ListImport Offres d'emploi
	Utilisations : ListReceipt Filtres
	Utilise : ListReceipt RuleSets

Préfixe de service	Actions
	Utilise : ListRecommendations
	Voir : ListSuppressed Destinations
	Utilise : ListTemplates
	Utilise : ListVerified EmailAddresses
	Utilise : PutAccount DedicatedIp WarmupAttributes
	Utilisation : PutAccount Détails
	Utilise : PutAccount SendingAttributes
	Utilise : PutAccount SuppressionAttributes
	Utilise : PutAccount VdmAttributes
	Utilise : PutConfiguration SetDelivery Options
	Utilise : PutConfiguration SetReputation Options
	Utilise : PutConfiguration SetSending Options
	Utilise : PutConfiguration SetSuppression Options
	Utilise : PutConfiguration SetTracking Options
	Utilise : PutConfiguration SetVdm Options
	Utilisations : PutDedicated IpIn Piscine
	Utilise : PutDedicated IpPool ScalingAttributes
	Utilisations : PutDedicated IpWarmup Attributs
	Utilise : PutDeliverability DashboardOption
	Utilise : PutEmail IdentityConfiguration SetAttributes
	Utilisations : PutEmail IdentityDkim Attributs

Préfixe de service	Actions
	Utilise : PutEmail IdentityDkim SigningAttributes
	Utilisations : PutEmail IdentityFeedback Attributs
	Utilise : PutEmail IdentityMail FromAttributes
	Utilisation : PutIdentity Politique
	Utilise : PutSuppressed Destination
	Utilise : ReorderReceipt RuleSet
	Utilise : SendBounce
	Utilise : SendCustom VerificationEmail
	Usages : SetActive ReceiptRule Set
	Utilise : SetIdentity DkimEnabled
	Utilisation : SetIdentity FeedbackForwarding Activé
	Utilise : SetIdentity HeadersIn NotificationsEnabled
	Utilisations : SetIdentity MailFrom Domain
	Utilise : SetIdentity NotificationTopic
	Utilise : SetReceipt RulePosition
	Utilise : TestRender EmailTemplate
	Utilisation : TestRender Modèle
	Utilise : UpdateAccount SendingEnabled
	Usages : UpdateConfiguration SetEvent Destination
	Utilise : UpdateConfiguration SetReputation MetricsEnabled
	Utilisation : UpdateConfiguration SetSending Activé

Préfixe de service	Actions
	Utilise : UpdateConfiguration SetTracking Options
	Utilise : UpdateContact
	Uses : UpdateContact Liste
	Utilisation : UpdateCustom VerificationEmail Modèle
	Utilise : UpdateEmail IdentityPolicy
	Utilisation : UpdateEmail Modèle
	Utilise : UpdateReceipt Règle
	Utilise : UpdateTemplate
	voir : VerifyDomain Dkim
	Utilisations : VerifyDomain Identité
	Utilisations : VerifyEmail Adresse
	Utilisations : VerifyEmail Identité

Préfixe de service	Actions
shield	Bouclier : Associated RT LogBucket bouclier : AssociateHealth Vérifiez bouclier : AssociateProactive EngagementDetails bouclier : CreateProtection bouclier : CreateProtection Groupe bouclier : CreateSubscription bouclier : DeleteProtection bouclier : DeleteProtection Groupe bouclier : DeleteSubscription bouclier : DescribeAttack shield : DescribeAttack Statistiques shield:DescribeDRTAccess bouclier : DescribeEmergency ContactSettings bouclier : DescribeProtection bouclier : DescribeProtection Groupe bouclier : DescribeSubscription bouclier : DisableApplication LayerAutomatic Réponse bouclier : DisableProactive Engagement Bouclier : Disassociated RT LogBucket shield:DisassociateDRTRole bouclier : DisassociateHealth Vérifiez

Préfixe de service	Actions
	bouclier : EnableApplication LayerAutomatic Réponse bouclier : EnableProactive Engagement bouclier : GetSubscription État bouclier : ListAttacks bouclier : ListProtection Groupes bouclier : ListProtections bouclier : ListResources InProtection Groupe bouclier : UpdateApplication LayerAutomatic Réponse bouclier : UpdateEmergency ContactSettings bouclier : UpdateProtection Groupe bouclier : UpdateSubscription

Préfixe de service	Actions
signer	signataire : AddProfile Autorisation
	signataire : CancelSigning Profil
	signataire : DescribeSigning Job
	signataire : GetRevocation Statut
	signataire : GetSigning Platform
	signataire : GetSigning Profil
	signataire : ListProfile Autorisations
	signataire : ListSigning Jobs
	signataire : ListSigning Platforms
	signataire : ListSigning Profiles
	signataire : PutSigning Profil
	signataire : RemoveProfile Autorisation
	signataire : RevokeSignature
	signataire : RevokeSigning Profil
	signataire : SignPayload
	signataire : StartSigning Job

Préfixe de service	Actions
simspaceweaver	Simspaceweaver : CreateSnapshot Simspaceweaver : DeleteApp Simspaceweaver : DeleteSimulation Simspaceweaver : DescribeApp Simspaceweaver : DescribeSimulation Simspaceweaver : ListApps Simspaceweaver : ListSimulations Simspaceweaver : StartApp Simspaceweaver : StartClock Simspaceweaver : StartSimulation Simspaceweaver : StopApp Simspaceweaver : StopClock Simspaceweaver : StopSimulation

Préfixe de service	Actions
sms	SMS : CreateApp sms : CreateReplication Job SMS : DeleteApp SMS : DeleteApp LaunchConfiguration SMS : DeleteApp ReplicationConfiguration SMS : DeleteApp ValidationConfiguration sms : DeleteReplication Job sms : DeleteServer Catalogue SMS : DisassociateConnector sms : GenerateChange Set SMS : GenerateTemplate SMS : GetApp SMS : GetApp LaunchConfiguration SMS : GetApp ReplicationConfiguration SMS : GetApp ValidationConfiguration SMS : GetApp ValidationOutput SMS : GetConnectors sms : GetReplication Offres d'emploi sms : GetReplication Fonctionne SMS : GetServers sms : ImportApp Catalogue

Préfixe de service	Actions
	sms : ImportServer Catalogue
	SMS : LaunchApp
	SMS : ListApps
	SMS : NotifyApp ValidationOutput
	SMS : PutApp LaunchConfiguration
	SMS : PutApp ReplicationConfiguration
	SMS : PutApp ValidationConfiguration
	sms : StartApp Réplication
	sms : StartOn DemandApp Réplication
	sms : StartOn DemandReplication Exécuter
	sms : StopApp Réplication
	SMS : TerminateApp
	SMS : UpdateApp
	sms : UpdateReplication Job

Préfixe de service	Actions
sms-voice	sms-voice : Configuration AssociateProtect sms-voice : Définir CreateConfiguration sms-voice : Destination CreateConfiguration SetEvent sms-voice : Destination CreateEvent SMS et voix : CreateOpt OutList SMS et voix : CreatePool sms-voice : Configuration CreateProtect SMS et voix : CreateRegistration sms-voice : Association CreateRegistration sms-voice : pièce jointe CreateRegistration sms-voice : Version CreateRegistration SMS et voix : CreateVerified DestinationNumber sms-voice : Configuration DeleteAccount DefaultProtect sms-voice : Définir DeleteConfiguration sms-voice : Destination DeleteConfiguration SetEvent SMS et voix : DeleteDefault MessageType SMS et voix : DeleteDefault SenderId sms-voice : Destination DeleteEvent SMS et voix : DeleteKeyword SMS et voix : DeleteMedia MessageSpend LimitOverride SMS et voix : DeleteOpted OutNumber

Préfixe de service	Actions
	SMS et voix : DeleteOpt OutList SMS et voix : DeletePool sms-voice : Configuration DeleteProtect SMS et voix : DeleteRegistration sms-voice : pièce jointe DeleteRegistration SMS et voix : DeleteText MessageSpend LimitOverride SMS et voix : DeleteVerified DestinationNumber SMS et voix : DeleteVoice MessageSpend LimitOverride sms-voice : Attributs DescribeAccount sms-voice : Limites DescribeAccount sms-voice : Ensembles DescribeConfiguration SMS et voix : DescribeKeywords SMS et voix : DescribeOpted OutNumbers SMS et voix : DescribeOpt OutLists sms-voice : Chiffres DescribePhone SMS et voix : DescribePools sms-voice : Configurations DescribeProtect sms-voice : Pièces jointes DescribeRegistration SMS et voix : DescribeRegistration FieldDefinitions SMS et voix : DescribeRegistration FieldValues SMS et voix : DescribeRegistrations

Préfixe de service	Actions
	SMS et voix : DescribeRegistration SectionDefinitions SMS et voix : DescribeRegistration TypeDefinitions sms-voice : Versions DescribeRegistration sms-voice : Identifiants DescribeSender sms-voice : Limites DescribeSpend SMS et voix : DescribeVerified DestinationNumbers sms-voice : Identité DisassociateOrigination sms-voice : Configuration DisassociateProtect sms-voice : Version DiscardRegistration sms-voice : Destinations GetConfiguration SetEvent SMS et voix : GetProtect ConfigurationCountry RuleSet sms-voice : Ensembles ListConfiguration SMS et voix : ListPool OriginationIdentities sms-voice : Associations ListRegistration SMS et voix : PutKeyword SMS et voix : PutOpted OutNumber sms-voice : Numéro ReleasePhone sms-voice : identifiant ReleaseSender sms-voice : Numéro RequestPhone sms-voice : identifiant RequestSender sms-voice : Code SendDestination NumberVerification

Préfixe de service	Actions
	sms-voice : Configuration SetAccount DefaultProtect SMS et voix : SetDefault MessageType SMS et voix : SetDefault SenderId SMS et voix : SetMedia MessageSpend LimitOverride SMS et voix : SetText MessageSpend LimitOverride SMS et voix : SetVoice MessageSpend LimitOverride sms-voice : Version SubmitRegistration sms-voice : Destination UpdateConfiguration SetEvent sms-voice : Destination UpdateEvent sms-voice : Numéro UpdatePhone SMS et voix : UpdatePool sms-voice : Configuration UpdateProtect SMS et voix : UpdateProtect ConfigurationCountry RuleSet sms-voice : identifiant UpdateSender

Préfixe de service	Actions
snowball	boule de neige : CancelCluster
	boule de neige : CancelJob
	boule de neige : CreateAddress
	boule de neige : CreateCluster
	boule de neige : CreateJob
	boule de neige : CreateLong TermPricing
	boule de neige : CreateReturn ShippingLabel
	boule de neige : DescribeAddress
	boule de neige : DescribeAddresses
	boule de neige : DescribeCluster
	boule de neige : DescribeJob
	boule de neige : DescribeReturn ShippingLabel
	boule de neige : Manifest GetJob
	boule de neige : GetJob UnlockCode
	boule de neige : Utilisation GetSnowball
	snowball : Mises à jour GetSoftware
	boule de neige : ListCluster Offres d'emploi
	boule de neige : ListClusters
	boule de neige : Images ListCompatible
	boule de neige : ListJobs
	boule de neige : ListLong TermPricing

Préfixe de service	Actions
	boule de neige : Localisations ListPickup boule de neige : Versions ListService boule de neige : UpdateCluster boule de neige : UpdateJob boule de neige : UpdateJob ShipmentState boule de neige : UpdateLong TermPricing
sqs	sqs : AddPermission sqs : CancelMessage MoveTask sqs : CreateQueue sqs : DeleteQueue sqs : PurgeQueue sqs : RemovePermission sqs : Attributs SetQueue

Préfixe de service	Actions
ssm	ssm : article AssociateOps ItemRelated SMS : CancelCommand SMS : CancelMaintenance WindowExecution SMS : CreateActivation SMS : CreateAssociation ssm : Batch CreateAssociation SMS : CreateDocument ssm : Fenêtre CreateMaintenance ssm : article CreateOps ssm : Métadonnées CreateOps ssm : CreatePatch base de référence SMS : CreateResource DataSync SMS : DeleteActivation SMS : DeleteAssociation SMS : DeleteDocument SMS : DeleteInventory ssm : Fenêtre DeleteMaintenance ssm : article DeleteOps ssm : Métadonnées DeleteOps SMS : DeleteParameter SMS : DeleteParameters

Préfixe de service	Actions
	ssm : DeletePatch base de référence
	SMS : DeleteResource DataSync
	ssm : Politique DeleteResource
	ssm : Instance DeregisterManaged
	SMS : DeregisterPatch BaselineFor PatchGroup
	ssm : Fenêtre DeregisterTarget FromMaintenance
	ssm : Fenêtre DeregisterTask FromMaintenance
	SMS : DescribeActivations
	SMS : DescribeAssociation
	ssm : Exécutions DescribeAssociation
	SMS : DescribeAssociation ExecutionTargets
	ssm : Exécutions DescribeAutomation
	SMS : DescribeAutomation StepExecutions
	ssm : Correctifs DescribeAvailable
	SMS : DescribeDocument
	ssm : Paramètres DescribeDocument
	ssm : Autorisation DescribeDocument
	SMS : DescribeEffective InstanceAssociations
	SMS : DescribeEffective PatchesFor PatchBaseline
	SMS : DescribeInstance AssociationsStatus
	ssm : Informations DescribeInstance

Préfixe de service	Actions
	ssm : Correctifs DescribeInstance
	SMS : DescribeInstance PatchStates
	ssm : Groupe DescribeInstance PatchStates ForPatch
	ssm : Propriétés DescribeInstance
	ssm : Suppressions DescribeInventory
	SMS : DescribeMaintenance WindowExecutions
	SMS : DescribeMaintenance WindowExecution TaskInvocations
	ssm : Tâches DescribeMaintenance WindowExecution
	SSM : Windows DescribeMaintenance
	SMS : DescribeMaintenance WindowSchedule
	ssm : cible DescribeMaintenance WindowsFor
	SMS : DescribeMaintenance WindowTargets
	SMS : DescribeMaintenance WindowTasks
	ssm : Objets DescribeOps
	SMS : DescribeParameters
	ssm : DescribePatch Lignes de base
	ssm : Groupes DescribePatch
	SMS : DescribePatch GroupState
	ssm : Propriétés DescribePatch
	SMS : DescribeSessions
	ssm : article DisassociateOps ItemRelated

Préfixe de service	Actions
	ssm : Exécution GetAutomation
	ssm : État GetCalendar
	ssm : Invocation GetCommand
	ssm : État GetConnection
	SMS : GetDefault PatchBaseline
	SMS : GetDeployable PatchSnapshot ForInstance
	SMS : GetDocument
	SMS : GetInventory
	ssm : Schéma GetInventory
	ssm : Fenêtre GetMaintenance
	SMS : GetMaintenance WindowExecution
	ssm : Tâche GetMaintenance WindowExecution
	SMS : GetMaintenance WindowExecution TaskInvocation
	SMS : GetMaintenance WindowTask
	ssm : article GetOps
	ssm : Métadonnées GetOps
	ssm : Résumé GetOps
	SMS : GetParameter
	ssm : Histoire GetParameter
	SMS : GetParameters
	SMS : GetParameters ByPath

Préfixe de service	Actions
	ssm : GetPatch base de référence
	SMS : GetPatch BaselineFor PatchGroup
	ssm : Politiques GetResource
	ssm : Réglage GetService
	ssm : Version LabelParameter
	SMS : ListAssociations
	ssm : Versions ListAssociation
	ssm : Invocations ListCommand
	SMS : ListCommands
	ssm : Objets ListCompliance
	ssm : Résumés ListCompliance
	SMS : ListDocument MetadataHistory
	SMS : ListDocuments
	ssm : Versions ListDocument
	ssm : Associations ListInstance
	ssm : Entrées ListInventory
	SMS : ListOps ItemEvents
	ssm : Objets ListOps ItemRelated
	ssm : Métadonnées ListOps
	SMS : ListResource ComplianceSummaries
	SMS : ListResource DataSync

Préfixe de service	Actions
	ssm : Autorisation ModifyDocument
	ssm : Objets PutCompliance
	SMS : PutInventory
	SMS : PutParameter
	ssm : Politique PutResource
	SMS : RegisterDefault PatchBaseline
	ssm : Instance RegisterManaged
	SMS : RegisterPatch BaselineFor PatchGroup
	ssm : Fenêtre RegisterTarget WithMaintenance
	ssm : Fenêtre RegisterTask WithMaintenance
	ssm : Réglage ResetService
	SMS : ResumeSession
	ssm : Signal SendAutomation
	SMS : SendCommand
	ssm : Une fois StartAssociations
	ssm : Exécution StartAutomation
	SMS : StartChange RequestExecution
	SMS : StartSession
	ssm : Exécution StopAutomation
	SMS : TerminateSession
	ssm : Version UnlabelParameter

Préfixe de service	Actions
	SMS : UpdateAssociation
	ssm : État UpdateAssociation
	SMS : UpdateDocument
	SMS : UpdateDocument DefaultVersion
	ssm : Métadonnées UpdateDocument
	ssm : Informations UpdateInstance
	ssm : Fenêtre UpdateMaintenance
	SMS : UpdateMaintenance WindowTarget
	SMS : UpdateMaintenance WindowTask
	SMS : UpdateManaged InstanceRole
	ssm : article UpdateOps
	ssm : Métadonnées UpdateOps
	ssm : UpdatePatch base de référence
	SMS : UpdateResource DataSync
	ssm : Réglage UpdateService

Préfixe de service	Actions
ssm-incidents	Incidents SMS : BatchGet IncidentFindings ssm-incidents : Set CreateReplication ssm-incidents : Planifier CreateResponse ssm-incidents : Événement CreateTimeline ssm-incidents : enregistrement DeleteIncident ssm-incidents : Set DeleteReplication ssm-incidents : Politique DeleteResource ssm-incidents : Planifier DeleteResponse ssm-incidents : Événement DeleteTimeline ssm-incidents : enregistrement GetIncident ssm-incidents : Set GetReplication ssm-incidents : Politiques GetResource ssm-incidents : Planifier GetResponse ssm-incidents : Événement GetTimeline ssm-incidents : résultats ListIncident ssm-incidents : Enregistrements ListIncident ssm-incidents : objets ListRelated ssm-incidents : Ensembles ListReplication ssm-incidents : Plans ListResponse ssm-incidents : Événements ListTimeline ssm-incidents : Politique PutResource

Préfixe de service	Actions
	Incidents SMS : StartIncident ssm-incidents : Protection UpdateDeletion ssm-incidents : enregistrement UpdateIncident ssm-incidents : objets UpdateRelated ssm-incidents : Set UpdateReplication ssm-incidents : Planifier UpdateResponse ssm-incidents : Événement UpdateTimeline

Préfixe de service	Actions
ssm-sap	SSM-SAP : BackupDatabase
	ssm-sap : Autorisation DeleteResource
	SSM-SAP : DeregisterApplication
	SSM-SAP : GetApplication
	SSM-SAP : GetComponent
	SSM-SAP : GetDatabase
	SSM-SAP : GetOperation
	ssm-sap : Autorisation GetResource
	SSM-SAP : ListApplications
	SSM-SAP : ListComponents
	SSM-SAP : ListDatabases
	ssm-sap : Événements ListOperation
	SSM-SAP : ListOperations
	ssm-sap : Autorisation PutResource
	SSM-SAP : RegisterApplication
	SSM-SAP : RestoreDatabase
	SSM-SAP : StartApplication
	ssm-sap : Actualiser StartApplication
	SSM-SAP : StopApplication
	ssm-sap : Paramètres UpdateApplication
	SSM-SAP : Mettre à jour Hana BackupSettings

Préfixe de service	Actions
states	états : CreateActivity états : CreateState Machine états : CreateState MachineAlias états : DeleteActivity états : DeleteState Machine états : DeleteState MachineAlias états : DeleteState MachineVersion états : DescribeActivity états : DescribeExecution états : DescribeMap Run états : DescribeState Machine états : DescribeState MachineAlias états : DescribeState MachineFor Exécution États : GetExecution Histoire états : ListActivities états : ListExecutions États : ListMap Runs états : ListState MachineAliases États : ListState Machines états : ListState MachineVersions états : SendTask Défaillance

Préfixe de service	Actions
	états : SendTask Heartbeat États : SendTask Succès états : StartExecution états : StopExecution états : UpdateMap Run états : UpdateState Machine états : UpdateState MachineAlias états : ValidateState MachineDefinition
sts	sets : AssumeRole sets : AssumeRole WithSAML sets : AssumeRole WithWeb Identité sts : DecodeAuthorization Message sets : GetAccess KeyInfo sets : GetCaller Identité sets : GetFederation Token sets : GetSession Token

Préfixe de service	Actions
swf	swf : Type DeprecateActivity swf : DeprecateDomain swf : Type DeprecateWorkflow swf : Type DescribeActivity swf : DescribeDomain swf : Type DescribeWorkflow swf : Types ListActivity swf : ListDomains swf : Types ListWorkflow swf : Type RegisterActivity swf : RegisterDomain swf : Type RegisterWorkflow swf : Type UndeprecateActivity swf : UndeprecateDomain swf : Type UndeprecateWorkflow

Préfixe de service	Actions
synthetics	synthétiques : AssociateResource
	synthétiques : CreateCanary
	synthétiques : CreateGroup
	synthétiques : DeleteCanary
	synthétiques : DeleteGroup
	synthétiques : DescribeCanaries
	synthétiques : DescribeCanaries LastRun
	synthétiques : Versions DescribeRuntime
	synthétiques : DisassociateResource
	synthétiques : GetCanary
	synthétiques : Runs GetCanary
	synthétiques : GetGroup
	synthétiques : Groupes ListAssociated
	synthétiques : Ressources ListGroup
	synthétiques : ListGroups
	synthétiques : StartCanary
	synthétiques : StopCanary
	synthétiques : UpdateCanary

Préfixe de service	Actions
balise	tag : DescribeReport Création tag : GetCompliance Résumé étiquette : GetResources tag : StartReport Création

Préfixe de service	Actions
textract	extrait : AnalyzeDocument
	extrait : AnalyzeExpense
	textract:AnalyzeID
	extrait : CreateAdapter
	extrait : Version CreateAdapter
	extrait : DeleteAdapter
	extrait : Version DeleteAdapter
	textract : DetectDocument Texte
	extrait : GetAdapter
	extrait : Version GetAdapter
	textract : Analyse GetDocument
	extrait : GetDocument TextDetection
	textract : Analyse GetExpense
	textract : Analyse GetLending
	extrait : GetLending AnalysisSummary
	extrait : ListAdapters
	extrait : Versions ListAdapter
	textract : Analyse StartDocument
	extrait : StartDocument TextDetection
	textract : Analyse StartExpense
	textract : Analyse StartLending

Préfixe de service	Actions
	extrait : UpdateAdapter

Préfixe de service	Actions
timestream	diffusion chronologique : CancelQuery diffusion chronologique : CreateDatabase timestream : Requête CreateScheduled diffusion chronologique : CreateTable diffusion chronologique : DeleteDatabase timestream : Requête DeleteScheduled diffusion chronologique : DeleteTable timestream : Réglages DescribeAccount diffusion chronologique : DescribeDatabase timestream : Requête DescribeScheduled diffusion chronologique : DescribeTable timestream : Requête ExecuteScheduled diffusion temporelle : ListBatch LoadTasks diffusion chronologique : ListDatabases timestream : requêtes ListScheduled diffusion chronologique : ListTables diffusion chronologique : PrepareQuery timestream : Réglages UpdateAccount diffusion chronologique : UpdateDatabase timestream : Requête UpdateScheduled diffusion chronologique : UpdateTable

Préfixe de service	Actions
tnb	tnb : CancelSol NetworkOperation
	tnb : CreateSol FunctionPackage
	tnb : CreateSol NetworkInstance
	tnb : CreateSol NetworkPackage
	tnb : DeleteSol FunctionPackage
	tnb : DeleteSol NetworkInstance
	tnb : DeleteSol NetworkPackage
	tnb : GetSol FunctionInstance
	tnb : GetSol FunctionPackage
	tnb : Contenu GetSol FunctionPackage
	tnb : Descripteur GetSol FunctionPackage
	tnb : GetSol NetworkInstance
	tnb : GetSol NetworkOperation
	tnb : GetSol NetworkPackage
	tnb : Contenu GetSol NetworkPackage
	tnb : Descripteur GetSol NetworkPackage
	tnb : InstantiateSol NetworkInstance
	tnb : ListSol FunctionInstances
	tnb : ListSol FunctionPackages
	tnb : ListSol NetworkInstances
	tnb : ListSol NetworkOperations

Préfixe de service	Actions
	tnb : ListSol NetworkPackages
	tnb : Contenu PutSol FunctionPackage
	tnb : Contenu PutSol NetworkPackage
	tnb : TerminateSol NetworkInstance
	tnb : UpdateSol FunctionPackage
	tnb : UpdateSol NetworkInstance
	tnb : UpdateSol NetworkPackage
	tnb : Contenu ValidateSol FunctionPackage
	tnb : Contenu ValidateSol NetworkPackage

Préfixe de service	Actions
transcribe	transcrire : CreateCall AnalyticsCategory transcrire : Modèle CreateLanguage transcrire : Vocabulaire CreateMedical transcrire : CreateVocabulary transcrire : Filtre CreateVocabulary transcrire : DeleteCall AnalyticsCategory transcrire : DeleteCall AnalyticsJob transcrire : Modèle DeleteLanguage transcrire : DeleteMedical ScribeJob transcrire : DeleteMedical TranscriptionJob transcrire : Vocabulaire DeleteMedical transcrire : Job DeleteTranscription transcrire : DeleteVocabulary transcrire : Filtre DeleteVocabulary transcrire : Modèle DescribeLanguage transcrire : GetCall AnalyticsCategory transcrire : GetCall AnalyticsJob transcrire : GetMedical ScribeJob transcrire : GetMedical TranscriptionJob transcrire : Vocabulaire GetMedical transcrire : Job GetTranscription

Préfixe de service	Actions
	transcrire : GetVocabulary transcrire : Filtre GetVocabulary transcrire : ListCall AnalyticsCategories transcrire : ListCall AnalyticsJobs transcrire : Modèles ListLanguage transcrire : ListMedical ScribeJobs transcrire : ListMedical TranscriptionJobs transcrire : Vocabulaires ListMedical transcrire : Jobs ListTranscription transcrire : ListVocabularies transcrire : Filtres ListVocabulary transcrire : StartCall AnalyticsJob transcrire : StartCall AnalyticsStream Transcription transcrire : Socket StartCall AnalyticsStream TranscriptionWeb transcrire : StartMedical ScribeJob transcrire : StartMedical StreamTranscription transcrire : StartMedical StreamTranscription WebSocket transcrire : StartMedical TranscriptionJob transcrire : StartStream Transcription transcrire : Socket StartStream TranscriptionWeb transcrire : Job StartTranscription

Préfixe de service	Actions
	transcrire : UpdateCall AnalyticsCategory transcrire : Vocabulaire UpdateMedical transcrire : UpdateVocabulary transcrire : Filtre UpdateVocabulary

Préfixe de service	Actions
transfert	transfert : CreateAccess
	transfert : CreateAgreement
	transfert : CreateConnector
	transfert : CreateProfile
	transfert : CreateServer
	transfert : CreateUser
	transfert : CreateWorkflow
	transfert : DeleteAccess
	transfert : DeleteAgreement
	transfert : DeleteCertificate
	transfert : DeleteConnector
	transfert : DeleteHost clé
	transfert : DeleteProfile
	transfert : DeleteServer
	transfert : DeleteSsh PublicKey
	transfert : DeleteUser
	transfert : DeleteWorkflow
	transfert : DescribeAccess
	transfert : DescribeAgreement
	transfert : DescribeCertificate
	transfert : DescribeConnector

Préfixe de service	Actions
	transfert : DescribeExecution
	transfert : DescribeHost clé
	transfert : DescribeProfile
	transfert : DescribeSecurity Politique
	transfert : DescribeServer
	transfert : DescribeUser
	transfert : DescribeWorkflow
	transfert : ImportCertificate
	transfert : ImportHost clé
	transfert : ImportSsh PublicKey
	transfert : ListAccesses
	transfert : ListCertificates
	transfert : ListConnectors
	transfert : ListExecutions
	transfert : ListHost Clés
	transfert : ListProfiles
	transfert : ListSecurity Politiques
	transfert : ListServers
	transfert : ListUsers
	transfert : ListWorkflows
	transfert : SendWorkflow StepState

Préfixe de service	Actions
	transfert : StartDirectory Listing
	transfert : StartFile Transfert
	transfert : StartServer
	transfert : StopServer
	transfert : TestConnection
	transfert : TestIdentity fournisseur
	transfert : UpdateAccess
	transfert : UpdateAgreement
	transfert : UpdateCertificate
	transfert : UpdateConnector
	transfert : UpdateHost clé
	transfert : UpdateProfile
	transfert : UpdateServer
	transfert : UpdateUser

Préfixe de service	Actions
translate	traduire : CreateParallel Data traduire : DeleteParallel Data traduire : DeleteTerminology traduire : DescribeText TranslationJob traduire : GetParallel Data traduire : GetTerminology traduire : ImportTerminology traduire : ListLanguages traduire : ListParallel Data traduire : ListTerminologies traduire : ListText TranslationJobs traduire : StartText TranslationJob traduire : StopText TranslationJob traduire : TranslateDocument traduire : TranslateText traduire : UpdateParallel Data

Préfixe de service	Actions
voiceid	identifiant vocal : AssociateFraudster
	identifiant vocal : CreateDomain
	identifiant vocal : CreateWatchlist
	identifiant vocal : DeleteDomain
	identifiant vocal : DeleteFraudster
	identifiant vocal : DeleteSpeaker
	identifiant vocal : DeleteWatchlist
	identifiant vocal : DescribeDomain
	identifiant vocal : DescribeFraudster
	identifiant vocal : DescribeFraudster RegistrationJob
	identifiant vocal : DescribeSpeaker
	identifiant vocal : DescribeSpeaker EnrollmentJob
	identifiant vocal : DescribeWatchlist
	identifiant vocal : DisassociateFraudster
	identifiant vocal : EvaluateSession
	identifiant vocal : ListDomains
	identifiant vocal : ListFraudster RegistrationJobs
	identifiant vocal : ListFraudsters
	identifiant vocal : ListSpeaker EnrollmentJobs
	identifiant vocal : ListSpeakers
	identifiant vocal : ListWatchlists

Préfixe de service	Actions
	voiceid : haut-parleur OptOut identifiant vocal : StartFraudster RegistrationJob identifiant vocal : StartSpeaker EnrollmentJob identifiant vocal : UpdateDomain identifiant vocal : UpdateWatchlist

Préfixe de service	Actions
vpc-lattice	réseau VPC : CreateAccess LogSubscription réseau VPC : CreateListener réseau VPC : CreateRule réseau VPC : CreateService vpc-lattice : Réseau CreateService vpc-lattice : Association CreateService NetworkService vpc-lattice : Association CreateService NetworkVpc vpc-lattice : Groupe CreateTarget réseau VPC : DeleteAccess LogSubscription vpc-lattice : Politique DeleteAuth réseau VPC : DeleteListener vpc-lattice : Politique DeleteResource réseau VPC : DeleteRule réseau VPC : DeleteService vpc-lattice : Réseau DeleteService vpc-lattice : Association DeleteService NetworkService vpc-lattice : Association DeleteService NetworkVpc vpc-lattice : Groupe DeleteTarget réseau VPC : DeregisterTargets réseau VPC : GetAccess LogSubscription vpc-lattice : Politique GetAuth

Préfixe de service	Actions
	réseau VPC : GetListener vpc-lattice : Politique GetResource réseau VPC : GetRule réseau VPC : GetService vpc-lattice : Réseau GetService vpc-lattice : Association GetService NetworkService vpc-lattice : Association GetService NetworkVpc vpc-lattice : Groupe GetTarget réseau VPC : ListAccess LogSubscriptions réseau VPC : ListListeners réseau VPC : ListRules vpc-lattice : Réseaux ListService vpc-lattice : Associations ListService NetworkService vpc-lattice : Associations ListService NetworkVpc réseau VPC : ListServices vpc-lattice : Groupes ListTarget réseau VPC : ListTargets vpc-lattice : Politique PutAuth vpc-lattice : Politique PutResource réseau VPC : RegisterTargets réseau VPC : UpdateAccess LogSubscription

Préfixe de service	Actions
	réseau VPC : UpdateListener réseau VPC : UpdateRule réseau VPC : UpdateService vpc-lattice : Réseau UpdateService vpc-lattice : Association UpdateService NetworkVpc vpc-lattice : Groupe UpdateTarget

Préfixe de service	Actions
wafv2	wafv2 : ACL AssociateWeb
	wafv2 : CheckCapacity
	wafv2:CreateAPIKey
	wafv2:CreateIPSet
	wafv2 : CreateRegex PatternSet
	wafv2 : Groupe CreateRule
	wafv2 : ACL CreateWeb
	WAF v2 : Supprimer la clé API
	wafv2 : Groupes DeleteFirewall ManagerRule
	wafv2:DeleteIPSet
	wafv2 : Configuration DeleteLogging
	wafv2 : Politique DeletePermission
	wafv2 : DeleteRegex PatternSet
	wafv2 : Groupe DeleteRule
	wafv2 : ACL DeleteWeb
	wafv2 : DescribeAll ManagedProducts
	wafv2 : Vendeur DescribeManaged ProductsBy
	wafv2 : DescribeManaged RuleGroup
	wafv2 : ACL DisassociateWeb
	wafv2 : URL GenerateMobile SdkRelease
	wafv2 : clé API GetDecrypted

Préfixe de service	Actions
	wafv2:GetIPSet
	wafv2 : Configuration GetLogging
	wafv2 : GetManaged RuleSet
	wafv2 : GetMobile SdkRelease
	wafv2 : Politique GetPermission
	wafv2 : GetRate BasedStatement ManagedKeys
	wafv2 : GetRegex PatternSet
	wafv2 : Groupe GetRule
	wafv2 : Demandes GetSampled
	wafv2 : ACL GetWeb ForResource
	wafv2:ListAPIKeys
	wafv2 : Groupes ListAvailable ManagedRule
	wafv2 : ListAvailable ManagedRule GroupVersions
	wafv2:ListIPSets
	wafv2 : Configurations ListLogging
	wafv2 : ListManaged RuleSets
	wafv2 : ListMobile SdkReleases
	wafv2 : ListRegex PatternSets
	wafv2 : ACL ListResources ForWeb
	wafv2 : Groupes ListRule
	wafv2 : ACL ListWeb

Préfixe de service	Actions
	wafv2 : Configuration PutLogging wafv2 : Versions PutManaged RuleSet wafv2 : Politique PutPermission wafv2:UpdateIPSet wafv2 : Date UpdateManaged RuleSet VersionExpiry wafv2 : UpdateRegex PatternSet wafv2 : Groupe UpdateRule wafv2 : ACL UpdateWeb

Préfixe de service	Actions
wellarchitected	bien conçu : AssociateLenses bien conçu : AssociateProfiles wellarchitected : Partager CreateLens wellarchitected : Version CreateLens bien conçu : CreateMilestone bien conçu : CreateProfile wellarchitected : Partager CreateProfile wellarchitected : Modèle CreateReview bien conçu : CreateWorkload wellarchitected : Partager CreateWorkload bien conçu : DeleteLens wellarchitected : Partager DeleteLens bien conçu : DeleteProfile wellarchitected : Partager DeleteProfile wellarchitected : Modèle DeleteReview wellarchitected : Partager DeleteTemplate bien conçu : DeleteWorkload wellarchitected : Partager DeleteWorkload bien conçu : DisassociateLenses bien conçu : DisassociateProfiles bien conçu : ExportLens

Préfixe de service	Actions
	bien conçu : GetAnswer wellarchitected : Rapport GetConsolidated wellarchitected : Paramètres GetGlobal bien conçu : GetLens wellarchitected : Critique GetLens bien conçu : GetLens ReviewReport bien conçu : GetLens VersionDifference bien conçu : GetMilestone bien conçu : GetProfile wellarchitected : Modèle GetProfile wellarchitected : Modèle GetReview bien conçu : GetReview TemplateAnswer wellarchitected : Critique GetReview TemplateLens bien conçu : GetWorkload bien conçu : ImportLens bien conçu : ListAnswers wellarchitected : Détails ListCheck wellarchitected : Résumés ListCheck bien conçu : ListLenses bien conçu : ListLens ReviewImprovements wellarchitected : Commentaires ListLens

Préfixe de service	Actions
	wellarchitected : Shares ListLens bien conçu : ListMilestones bien conçu : ListNotifications wellarchitected : Notifications ListProfile bien conçu : ListProfiles wellarchitected : Shares ListProfile bien conçu : ListReview TemplateAnswers wellarchitected : Modèles ListReview wellarchitected : Invitations ListShare wellarchitected : Shares ListTemplate bien conçu : ListWorkloads wellarchitected : Shares ListWorkload bien conçu : UpdateAnswer wellarchitected : Paramètres UpdateGlobal bien conçu : UpdateIntegration wellarchitected : Critique UpdateLens bien conçu : UpdateProfile wellarchitected : Modèle UpdateReview wellarchitected : Critique UpdateReview TemplateLens wellarchitected : Invitation UpdateShare bien conçu : UpdateWorkload

Préfixe de service	Actions
	wellarchitected : Partager UpdateWorkload wellarchitected : Critique UpgradeLens wellarchitected : Version UpgradeProfile wellarchitected : Critique UpgradeReview TemplateLens

Préfixe de service	Actions
wisdom	sagesse : CreateAssistant
	sagesse : CreateAssistant Association
	sagesse : CreateContent
	sagesse : CreateKnowledge Base
	sagesse : CreateQuick Réponse
	sagesse : CreateSession
	sagesse : DeleteAssistant
	sagesse : DeleteAssistant Association
	sagesse : DeleteContent
	sagesse : DeleteImport Job
	sagesse : DeleteKnowledge Base
	sagesse : DeleteQuick Réponse
	sagesse : GetAssistant
	sagesse : GetAssistant Association
	sagesse : GetContent
	sagesse : GetContent Résumé
	sagesse : GetImport Job
	sagesse : GetKnowledge Base
	sagesse : GetRecommendations
	sagesse : GetSession
	sagesse : ListAssistant Associations

Préfixe de service	Actions
	sagesse : ListAssistants
	sagesse : ListContents
	sagesse : ListImport Emplois
	sagesse : ListKnowledge Bases
	sagesse : ListQuick Réponses
	sagesse : NotifyRecommendations Reçu
	sagesse : QueryAssistant
	sagesse : RemoveKnowledge BaseTemplate Uri
	sagesse : SearchContent
	sagesse : SearchQuick Réponses
	sagesse : SearchSessions
	wisdom : StartContent Télécharger
	sagesse : StartImport Job
	sagesse : UpdateContent
	sagesse : UpdateKnowledge BaseTemplate Uri
	sagesse : UpdateQuick Réponse

Préfixe de service	Actions
worklink	lien de travail : AssociateDomain lien de travail : AssociateWebsite AuthorizationProvider lien de travail : AssociateWebsite CertificateAuthority lien de travail : CreateFleet lien de travail : DeleteFleet lien de travail : DescribeAudit StreamConfiguration lien de travail : DescribeCompany NetworkConfiguration lien de travail : DescribeDevice lien de travail : DescribeDevice PolicyConfiguration lien de travail : DescribeDomain worklink : Métadonnées DescribeFleet lien de travail : DescribeIdentity ProviderConfiguration lien de travail : DescribeWebsite CertificateAuthority lien de travail : DisassociateDomain lien de travail : DisassociateWebsite AuthorizationProvider lien de travail : DisassociateWebsite CertificateAuthority lien de travail : ListDevices lien de travail : ListDomains lien de travail : ListFleets lien de travail : ListWebsite AuthorizationProviders lien de travail : ListWebsite CertificateAuthorities

Préfixe de service	Actions
	worklink : Accès RestoreDomain worklink : Accès RevokeDomain worklink : Utilisateur SignOut lien de travail : UpdateAudit StreamConfiguration lien de travail : UpdateCompany NetworkConfiguration lien de travail : UpdateDevice PolicyConfiguration worklink : Métadonnées UpdateDomain worklink : Métadonnées UpdateFleet lien de travail : UpdateIdentity ProviderConfiguration

Préfixe de service	Actions
espaces de travail	espaces de travail : AcceptAccount LinkInvitation
	espaces de travail : Alias AssociateConnection
	espaces de travail : Groupes AssociateIp
	espaces de travail : Application AssociateWorkspace
	espaces de travail : Image CopyWorkspace
	espaces de travail : Dans CreateConnect ClientAdd
	espaces de travail : Alias CreateConnection
	espaces de travail : Groupe CreateIp
	espaces de travail : CreateStandby Espaces de travail
	espaces de travail : CreateUpdated WorkspaceImage
	espaces de travail : CreateWorkspace Bundle
	espaces de travail : Image CreateWorkspace
	espaces de travail : CreateWorkspaces
	espaces de travail : Branding DeleteClient
	espaces de travail : Dans DeleteConnect ClientAdd
	espaces de travail : Alias DeleteConnection
	espaces de travail : Groupe DeleteIp
	espaces de travail : DeleteWorkspace Bundle
	espaces de travail : Image DeleteWorkspace
	espaces de travail : Applications DeployWorkspace
	espaces de travail : Répertoire DeregisterWorkspace

Préfixe de service	Actions
	espaces de travail : DescribeAccount
	espaces de travail : modifications DescribeAccount
	espaces de travail : Associations DescribeApplication
	espaces de travail : DescribeApplications
	espaces de travail : Associations DescribeBundle
	espaces de travail : Branding DescribeClient
	espaces de travail : propriétés DescribeClient
	espaces de travail : Ins DescribeConnect ClientAdd
	espaces de travail : Alias DescribeConnection
	espaces de travail : DescribeConnection AliasPermissions
	espaces de travail : Associations DescribeImage
	espaces de travail : Groupes DescribeIp
	espaces de travail : Associations DescribeWorkspace
	espaces de travail : DescribeWorkspace Bundles
	espaces de travail : Répertoires DescribeWorkspace
	espaces de travail : DescribeWorkspace ImagePermissions
	espaces de travail : DescribeWorkspaces
	espaces de travail : DescribeWorkspaces ConnectionStatus
	espaces de travail : DescribeWorkspace instantanés
	espaces de travail : Alias DisassociateConnection
	espaces de travail : Groupes DisassociateIp

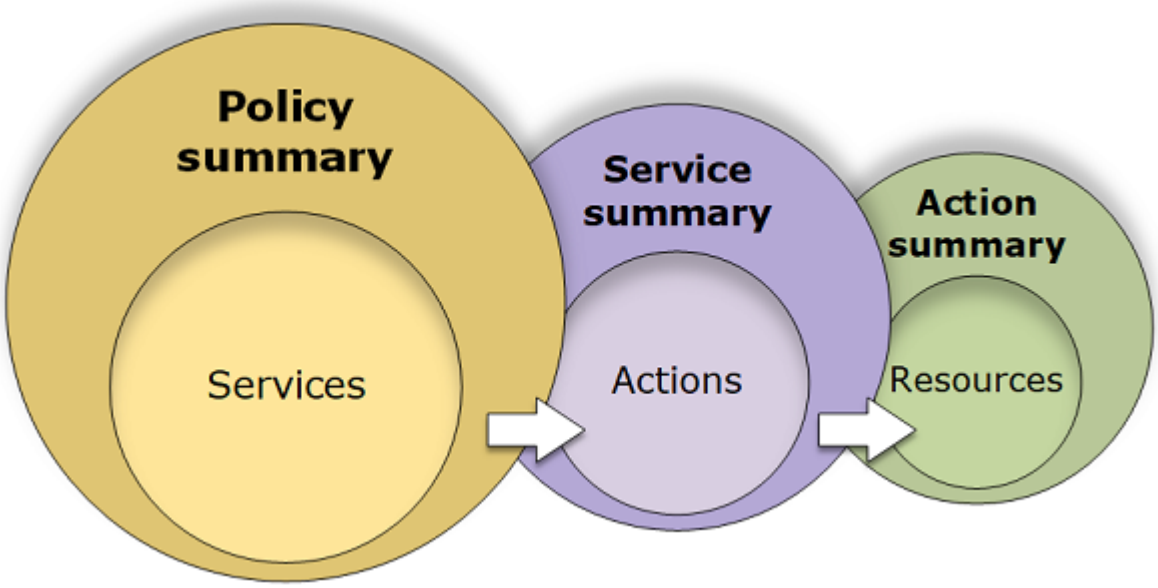
Préfixe de service	Actions
	espaces de travail : Application DisassociateWorkspace
	espaces de travail : Link GetAccount
	espaces de travail : Branding ImportClient
	espaces de travail : Image ImportWorkspace
	espaces de travail : liens ListAccount
	espaces de travail : Ranges ListAvailable ManagementCidr
	espaces de travail : MigrateWorkspace
	espaces de travail : ModifyAccount
	espaces de travail : propriétés ModifyCertificate BasedAuth
	espaces de travail : propriétés ModifyClient
	espaces de travail : propriétés ModifySaml
	espaces de travail : autorisations ModifySelfservice
	espaces de travail : ModifyWorkspace AccessProperties
	espaces de travail : ModifyWorkspace CreationProperties
	espaces de travail : propriétés ModifyWorkspace
	espaces de travail : État ModifyWorkspace
	espaces de travail : RebootWorkspaces
	espaces de travail : RebuildWorkspaces
	espaces de travail : Répertoire RegisterWorkspace
	espaces de travail : RejectAccount LinkInvitation
	espaces de travail : RestoreWorkspace

Préfixe de service	Actions
	espaces de travail : StartWorkspaces
	espaces de travail : StopWorkspaces
	espaces de travail : TerminateWorkspaces
	espaces de travail : Dans UpdateConnect ClientAdd
	espaces de travail : UpdateConnection AliasPermission
	espaces de travail : UpdateWorkspace Bundle
	espaces de travail : UpdateWorkspace ImagePermission

Préfixe de service	Actions
xray	radiographie : CreateGroup Radiographie : CreateSampling Règle radiographie : DeleteGroup Radiographie : Politique DeleteResource Radiographie : DeleteSampling Règle xray : Config GetEncryption radiographie : GetGroup radiographie : GetGroups radiographie : GetInsight xray : Événements GetInsight radiographie : GetInsight ImpactGraph radiographie : Résumés GetInsight Radiographie : Règles GetSampling xray : Politiques ListResource xray : Config PutEncryption Radiographie : Politique PutResource radiographie : UpdateGroup Radiographie : UpdateSampling Règle

Comprendre les autorisations accordées par une politique

La console IAM comprend des tables de récapitulatif de la politique qui présentent le niveau d'accès, les ressources et les conditions autorisées ou rejetées pour chaque service dans une politique. Les politiques sont résumées dans trois tables : récapitulatif de la politique, récapitulatif du service et récapitulatif de l'action. La table de récapitulatif de la politique comprend une liste de services. Choisissez un service pour voir le récapitulatif du service. Cette table récapitulative comprend une liste des actions et autorisations associées pour le service choisi. Vous pouvez choisir une action dans cette table pour afficher le récapitulatif de l'action. Cette table comprend une liste des ressources et conditions pour l'action choisie.



Vous pouvez afficher les récapitulatifs de politiques sur la page Utilisateurs ou Rôles pour toutes les politiques (gérées et en ligne) attachées à cet utilisateur. Affichez les récapitulatifs sur la page Politiques pour toutes les politiques gérées. Les politiques gérées incluent les politiques AWS gérées, les politiques AWS gérées des fonctions de travail et les politiques gérées par le client. Vous pouvez consulter les récapitulatifs de ces politiques sur la page Politiques qu'elles soient attachées à un utilisateur ou à une autre identité IAM.

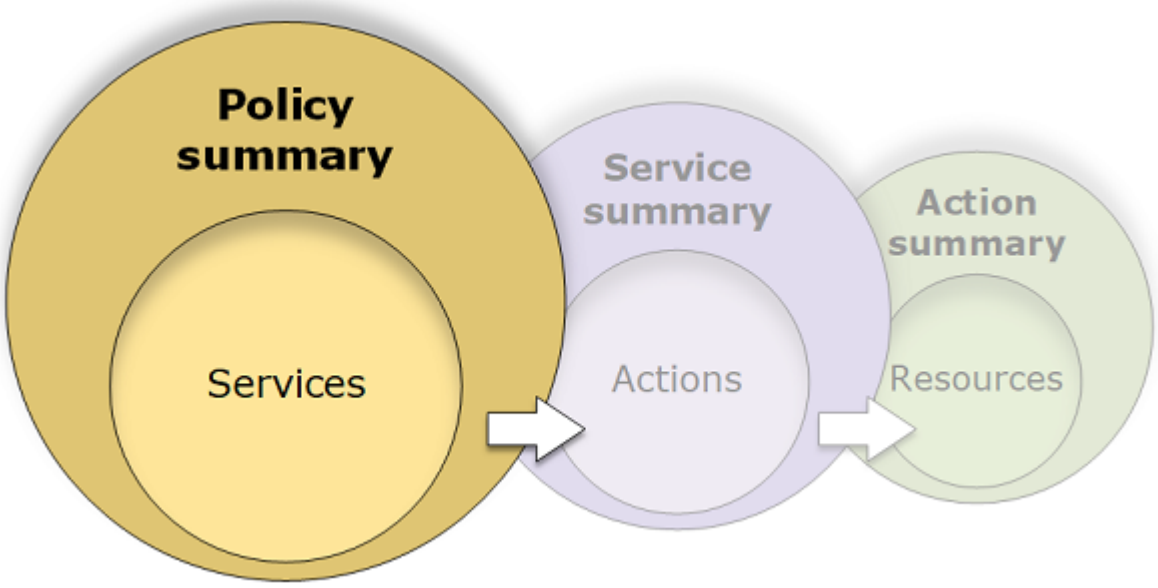
Vous pouvez utiliser les informations des récapitulatifs de la politique pour comprendre les autorisations accordées ou refusées par votre politique. Les récapitulatifs de la politique peuvent vous aider à résoudre les problèmes des politiques qui ne fournissent pas les autorisations que vous attendiez.

Rubriques

- Récapitulatif de la politique (liste des services)
- Récapitulatif du service (liste des actions)
- Récapitulatif de l'action (liste des ressources)
- Exemples de récapitulatifs de la politique

Récapitulatif de la politique (liste des services)

Les politiques sont résumées dans trois tables : récapitulatif de la politique, récapitulatif du service et récapitulatif de l'action. La table récapitulative de la politique comprend une liste des services et des résumés des autorisations définies par la politique choisie.



La table récapitulative de la politique est regroupée en une ou plusieurs sections Uncategorized services (Services non catégorisés), Explicit deny (Refus explicite) et Allow (Autoriser). Si la politique comprend un service qu'IAM ne reconnaît pas, le service est inclus dans la section Uncategorized services (Services non catégorisés) de la table. Si IAM reconnaît le service, celui-ci figure dans la section Explicit deny (Refus explicite) ou Allow (Autoriser) de la table, selon l'effet de la politique (`Deny` ou `Allow`).

Affichage des récapitulatifs de politique

Vous pouvez afficher les résumés de toutes les politiques attachées à un utilisateur en choisissant le nom de la politique dans l'onglet Autorisations de la page des détails de l'utilisateur. Vous pouvez afficher les résumés de toutes les politiques attachées à un rôle en choisissant le nom de la politique

dans l'onglet Autorisations de la page des détails du rôle. Vous pouvez afficher le récapitulatif des politiques gérées sur la page Politiques. Si votre politique ne comprend pas de récapitulatif de la politique, consultez Récapitulatif de politique manquant pour en comprendre la raison.

Pour afficher le récapitulatif de la stratégie sur la page Stratégies

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de la stratégie à afficher.
4. Sur la page Détails de la politique, affichez l'onglet Autorisations pour consulter le résumé de la politique.

Pour afficher le récapitulatif d'une politique attachée à un utilisateur

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Utilisateurs dans le panneau de navigation.
3. Dans la liste des utilisateurs, choisissez le nom de l'utilisateur dont vous souhaitez afficher la politique.
4. Sur la page Récapitulatif de l'utilisateur, affichez l'onglet Autorisations pour afficher la liste des stratégies attachées à l'utilisateur directement ou via un groupe.
5. Dans le tableau des stratégies dédiées à l'utilisateur, développez la ligne de la stratégie à afficher.

Pour afficher le récapitulatif d'une politique attachée à un rôle

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Roles (Rôles).
3. Dans la liste des rôles, choisissez le nom du rôle dont vous souhaitez afficher la politique.
4. Sur la page Récapitulatif du rôle, affichez l'onglet Autorisations pour afficher la liste des stratégies attachées au rôle.
5. Dans le tableau des stratégies dédiées au rôle, développez la ligne de la stratégie à afficher.

Modification des politiques pour corriger les avertissements

Lorsque vous consultez le récapitulatif d'une politique, vous pouvez détecter une faute de frappe ou remarquer que la politique ne fournit pas les autorisations que vous attendiez. Vous ne pouvez pas modifier le récapitulatif d'une politique directement. Toutefois, vous pouvez modifier une politique gérée par le client à l'aide de l'éditeur de politique visuel, qui détecte un grand nombre des mêmes erreurs et avertissements que les rapports de synthèse de politique. Vous pouvez ensuite consulter les modifications dans le résumé de politique pour confirmer que vous avez résolu tous les problèmes. Pour savoir comment modifier une politique en ligne, consulter the section called “Modification de politiques IAM”. Vous ne pouvez pas modifier les politiques AWS gérées.

Pour modifier une politique pour votre résumé de la politique à l'aide de l'option Visuel

1. Ouvrez le récapitulatif de politique comme expliqué dans les procédures précédentes.
2. Choisissez Modifier.

   Si vous vous trouvez sur la page Utilisateurs et que vous choisissez de modifier une stratégie gérée par le client qui est attachée à ce dernier, vous êtes redirigé vers la page Stratégies. Vous ne pouvez modifier les stratégies gérées par le client que sur la page Stratégies.
3. Choisissez l'option Visuel pour afficher la représentation visuelle modifiable de votre politique. IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel et de vous permettre de trouver et de résoudre plus facilement tous les problèmes. Les avertissements et messages d'erreur sur la page peuvent vous aider à résoudre tous les problèmes liés à votre politique. Pour plus d'informations sur les politiques de restructuration IAM, voir Restructuration de politique.
4. Modifiez votre politique et choisissez Suivant pour que vos modifications s'affichent dans le résumé de la politique. Si un problème persiste, choisissez Précédent pour retourner à l'écran de modification.
5. Choisissez Enregistrer les Modifications pour enregistrer vos Modifications.

Pour modifier une politique pour votre résumé de la politique à l'aide de l'option JSON

1. Ouvrez le récapitulatif de politique comme expliqué dans les procédures précédentes.
2. Vous pouvez utiliser les boutons Résumé et JSON pour comparer le résumé de la politique au document de politique JSON. Ces informations peuvent vous permettre de déterminer quelles lignes du document de politique vous souhaitez modifier.
3. Choisissez Modifier, puis l'option JSON pour modifier le document de politique JSON.

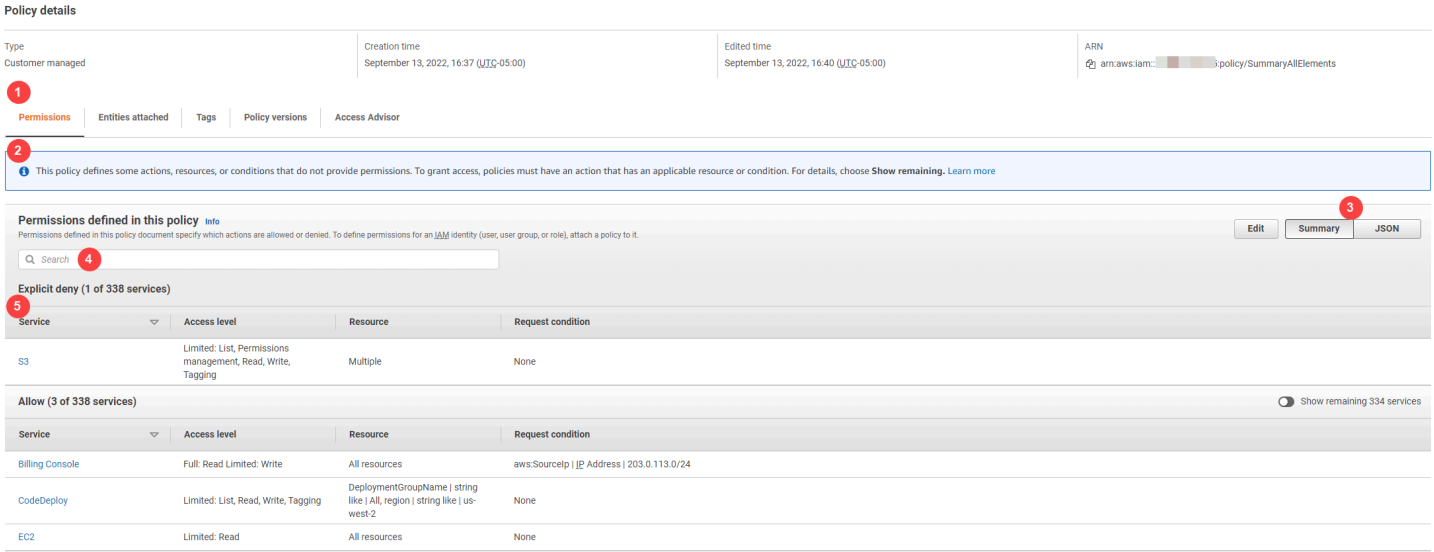
> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'option éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez Restructuration de politique.

Si vous vous trouvez sur la page Utilisateurs et que vous choisissez de modifier une stratégie gérée par le client qui est attachée à ce dernier, vous êtes redirigé vers la page Stratégies. Vous ne pouvez modifier les stratégies gérées par le client que sur la page Stratégies.

4. Modifiez votre politique. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant. Si un problème persiste, choisissez Précédent pour retourner à l'écran de modification.
5. Choisissez Enregistrer les Modifications pour enregistrer vos Modifications.

Présentation des éléments d'un récapitulatif de service

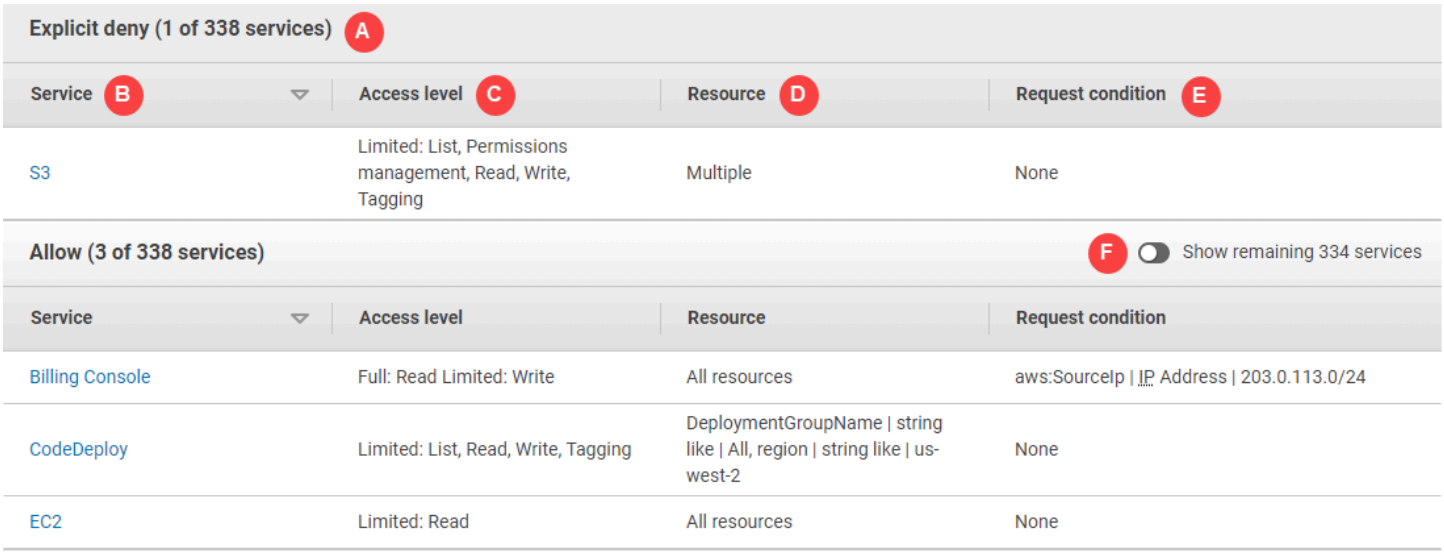
Dans l'exemple suivant de page détaillée d'une politique, la SummaryAllElementspolitique est une politique gérée (politique gérée par le client) attachée directement à l'utilisateur. Cette politique est développée pour afficher son récapitulatif.



Dans l'image précédente, le résumé de la politique est visible sur la page Politiques :

1. L'onglet Autorisations comprend les autorisations définies dans la politique.
2. Si la politique n'accorde pas les autorisations pour toutes les actions, ressources et conditions définies dans la politique, un avertissement ou une bannière d'erreur s'affiche dans la partie supérieure de la page. Puis le récapitulatif de politique inclut des détails relatifs au problème. Pour savoir comment les récapitulatifs de politique vous aident à comprendre les autorisations que votre politique vous accorde et à résoudre les problèmes les concernant, consultez the section called “Ma politique n'accorde pas les autorisations escomptées”.
3. Utilisez les boutons Résumé et JSON pour basculer entre le résumé de la politique et le document de politique JSON.
4. Utilisez la zone de Recherche pour réduire la liste des services et trouver un service spécifique.
5. La vue agrandie affiche des détails supplémentaires sur la SummaryAllElementspolitique.

L'image du tableau récapitulatif des politiques ci-dessous montre la SummaryAllElementsstratégie étendue sur la page des détails de la stratégie.



Dans l'image précédente, le résumé de la politique est visible sur la page Politiques :

A. En ce qui concerne les services qu'IAM reconnaît, il dispose les services selon si la politique autorise ou refuse explicitement l'utilisation du service. Dans cet exemple, la politique inclut une `Deny` déclaration pour le service Amazon S3 et `Allow` des instructions pour les services de facturation et Amazon EC2. CodeDeploy

B. Service : cette colonne répertorie les services définis dans la politique et présente des détails pour chaque service. Chaque nom de service dans la table récapitulative de la politique est un lien vers la table de récapitulatif du service, présenté dans la section Récapitulatif du service

(liste des actions). Dans cet exemple, les autorisations sont définies pour les services Amazon S3 CodeDeploy, Billing et Amazon EC2.

C. Niveau d'accès : cette colonne indique si les actions dans chaque niveau d'accès (`List`, `Read`, `Write`, `Permission Management` et `Tagging`) ont des autorisations `Full` ou `Limited` définies dans la politique. Pour plus d'informations et pour consulter des exemples de récapitulatif de niveau d'accès, consultez Comprendre les niveaux d'accès dans les résumés des politiques.

- Full access (Accès complet) : cette entrée indique que le service a accès à toutes les actions dans les quatre niveaux d'accès disponibles pour le service.
- Si l'entrée ne comprend pas l'Accès complet, le service a accès à certains actions, mais pas à l'ensemble des actions, pour le service. L'accès est alors défini en suivant les descriptions pour chacune des classifications du niveau d'accès (`List`, `Read`, `Write`, `Permission Management` et `Tagging`) :

  Complet : La politique fournit un accès complet à toutes les actions contenues dans chaque classification de niveau d'accès répertoriée. Dans cet exemple, la politique permet l'accès à toutes les actions `Read` de la facturation.

  Limité : La politique fournit un accès à une ou plusieurs actions, mais pas à l'ensemble des actions contenues dans chaque classification de niveau d'accès répertoriée. Dans cet exemple, la politique permet l'accès à certaines des actions `Write` de la facturation.

D. Resource (Ressource) : cette colonne présente les ressources que la politique définit pour chaque service.

- Multiple (Plusieurs) : la politique comprend plusieurs ressources, mails pas leur totalité, dans le service. Dans cet exemple, l'accès est explicitement refusé à plusieurs ressources Amazon S3.
- Toutes les ressources : la stratégie est définie pour toutes les ressources du service. Dans cet exemple, la politique autorise les actions répertoriées sur toutes les ressources de la facturation.
- Resource text (Texte de ressource) : la politique contient une ressource dans le service. Dans cet exemple, les actions répertoriées ne sont autorisées que sur la `DeploymentGroupName` CodeDeploy ressource. En fonction des informations que le service fournit à IAM, il est possible que vous voyiez un ARN ou le type de ressource défini.

> **Note**
>
> Cette colonne peut contenir une ressource provenant d'un service différent. Si l'instruction de politique qui inclut la ressource ne contient pas d'actions et de ressources provenant d'un même service, cela signifie que votre politique contient des ressources

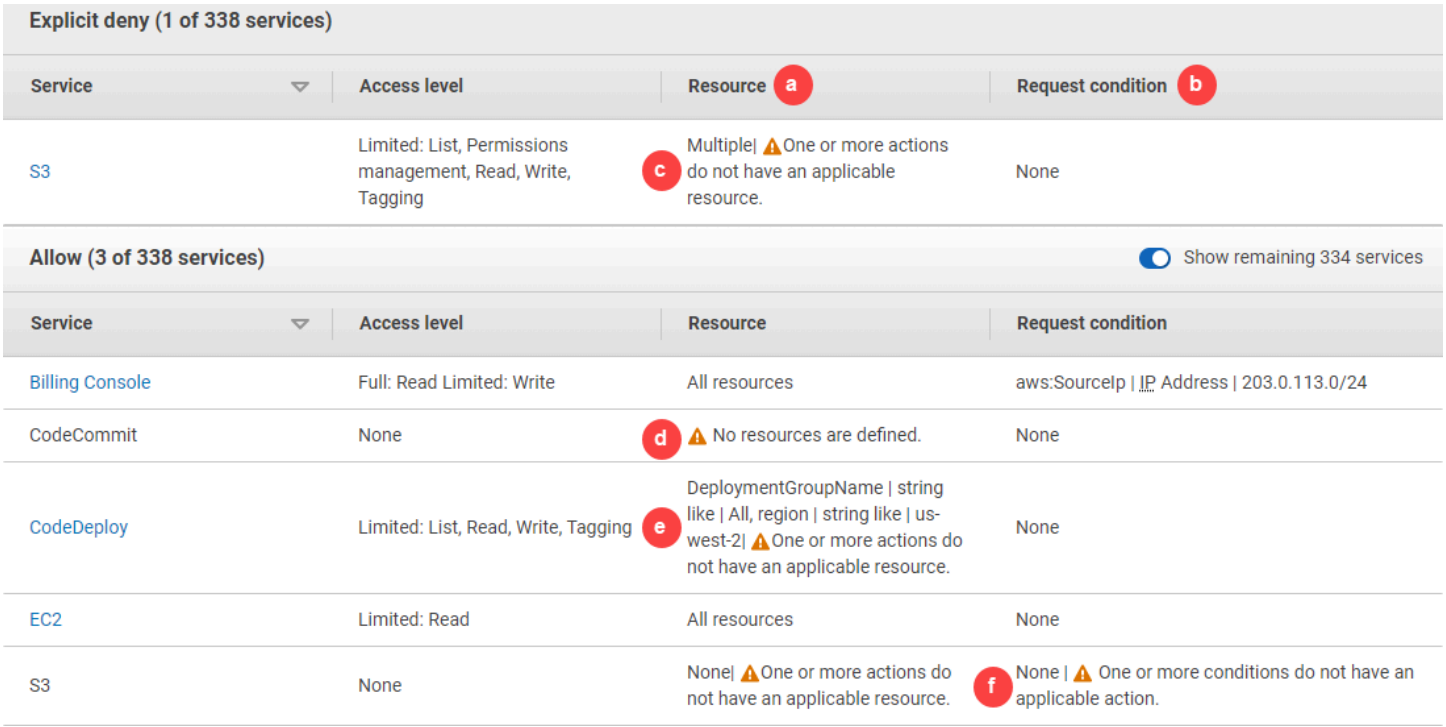
non appariées. IAM ne vous avertit pas que des ressources ne sont pas appariées lorsque vous créez une politique ou que vous affichez une politique dans le récapitulatif de la politique. Si cette colonne contient une ressource non appariée, recherchez d'éventuelles erreurs dans votre politique. Afin de mieux comprendre vos politiques, vous devez toujours les tester avec le simulateur de politique.

E. Request condition (Demander une condition) : cette colonne indique si les services ou actions associés aux ressources font l'objet de conditions.

- None (Aucune) : la politique ne contient aucune condition pour le service. Dans cet exemple, aucun condition n'est appliquée aux actions refusées dans le service Amazon S3.
- Condition text (Texte de condition) : la politique contient une condition pour le service. Dans cet exemple, les actions de Facturation énumérées sont autorisées uniquement si l'adresse IP de la source correspond à `203.0.113.0/24`.
- Multiple (Plusieurs) : la politique contient plusieurs conditions pour le service. Pour afficher chacune des conditions multiples de la politique, choisissez JSON pour afficher le document de politique.

F. Afficher les services restants : activez ce bouton pour développer la table afin d'inclure les services non définis par la politique. Ces services sont refusés implicitement (ou refusés par défaut) dans cette politique. Cependant, une instruction dans une autre politique peut tout de même autoriser ou refuser explicitement l'utilisation du service. Le récapitulatif de la politique résume les autorisations d'une seule politique. Pour savoir comment le AWS service décide si une demande donnée doit être autorisée ou refusée, voirLogique d'évaluation de politiques.

Lorsqu'une politique ou un élément dans la politique n'accorde pas d'autorisations, IAM fournit des avertissements et des informations supplémentaires dans le récapitulatif de politique. Le tableau récapitulatif de la politique suivant montre les services étendus Afficher les services restants sur la page des détails de la SummaryAllElementspolitique avec les avertissements possibles.



Explicit deny (1 of 338 services)

Service	Access level	Resource a	Request condition b
S3	Limited: List, Permissions management, Read, Write, Tagging	c Multiple| ⚠ One or more actions do not have an applicable resource.	None

Allow (3 of 338 services) — Show remaining 334 services

Service	Access level	Resource	Request condition
Billing Console	Full: Read Limited: Write	All resources	aws:SourceIp | IP Address | 203.0.113.0/24
CodeCommit	None	d ⚠ No resources are defined.	None
CodeDeploy	Limited: List, Read, Write, Tagging	e DeploymentGroupName | string like | All, region | string like | us-west-2| ⚠ One or more actions do not have an applicable resource.	None
EC2	Limited: Read	All resources	None
S3	None	None| ⚠ One or more actions do not have an applicable resource.	f None | ⚠ One or more conditions do not have an applicable action.

Dans l'image précédente, vous pouvez voir tous les services incluant des actions, ressources ou conditions définies sans autorisation :

a. Resource warnings (Avertissements relatifs aux ressources) : pour les services qui ne fournissent pas d'autorisations pour toutes les actions ou ressources incluses, vous voyez l'un des avertissements suivants dans la colonne Resource (Ressource) de la table :

   - ⚠ Aucune ressource n'est définie. – Cela signifie que le service a défini des actions, mais qu'aucune ressource prise en charge n'est incluse dans la politique.
   - ⚠ Une ou plusieurs actions n'ont pas de ressource applicable. – Cela signifie que le service a défini des actions, mais que certaines d'entre elles n'ont pas de ressource prise en charge.
   - ⚠ Une ou plusieurs ressources n'ont pas d'action applicable. – Cela signifie que le service a défini des ressources, mais que certaines d'entre elles n'ont pas d'action associée.

   Si un service comprend à la fois des actions qui n'ont pas de ressource applicable et des ressources qui ont une ressource applicable, alors seul l'avertissement Une ou plusieurs ressources n'ont pas d'action applicable est affiché. Ceci est dû au fait que lorsque vous affichez

le récapitulatif de ce service, les ressources qui ne s'appliquent à aucune action ne s'affichent pas. Pour l'action `ListAllMyBuckets`, cette politique inclut le dernier avertissement, car l'action ne prend pas en charge les autorisations au niveau des ressources, ni la clé de condition `s3:x-amz-acl`. Si vous corrigez le problème de ressource ou celui de condition, le problème restant s'affiche dans un avertissement détaillé.

b. Request condition warnings (Avertissements relatifs aux demandes de condition) : pour les services qui ne fournissent pas d'autorisations pour toutes les conditions incluses, vous voyez l'un des avertissements suivants dans la colonne Request condition (Demander une condition) de la table :

   - 

     Une ou plusieurs actions n'ont pas de condition applicable. – Cela signifie que le service a défini des actions, mais que certaines d'entre elles n'ont pas de condition prise en charge.

   - 

     Une ou plusieurs conditions n'ont pas d'action applicable. – Cela signifie que le service a défini des conditions, mais que certaines d'entre elles n'ont pas d'action associée.

c. Multiple (Plusieurs) |

   Une ou plusieurs actions n'ont pas de ressource applicable. – L'instruction `Deny` pour Amazon S3 inclut plusieurs ressources. Elle inclut également plusieurs actions ; certaines actions prennent en charge les ressources et d'autres non. Pour afficher cette politique, consultez the section called “Document de politique JSON SummaryAllElements”. Dans ce cas, la politique inclut toutes les actions d'Amazon S3, et seules les actions qui peuvent être effectuées sur un compartiment ou un objet dans un compartiment sont rejetées.

d. 

   No resources are defined (Aucune ressource n'est définie) : le service a défini des actions, mais aucune ressource prise en charge n'est incluse dans la politique, et par conséquent, le service ne fournit aucune autorisation. Dans ce cas, la politique inclut des CodeCommit actions mais aucune CodeCommit ressource.

e. DeploymentGroupName | string like | All, region | string like | us-west-2

   | Aucune ressource applicable n'est associée à une ou plusieurs actions. – Le service a une action définie et au moins une autre action qui n'a pas de ressource de support.

f. None |

Une ou plusieurs conditions n'ont pas d'action applicable. – Le service a au moins une clé de condition qui n'a pas d'action de support.

Document de politique JSON SummaryAllElements

La SummaryAllElementspolitique n'est pas destinée à être utilisée pour définir des autorisations dans votre compte. En fait, elle est incluse pour illustrer les erreurs et les avertissements que vous risquez de rencontrer lors de l'affichage d'un récapitulatif de politique.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "billing:Get*",
                "payments:List*",
                "payments:Update*",
                "account:Get*",
                "account:List*",
                "cur:GetUsage*"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action": [
                "s3:*"
            ],
            "Resource": [
                "arn:aws:s3:::customer",
                "arn:aws:s3:::customer/*"
```

```
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshots"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "codedploy:*",
                "codecommit:*"
            ],
            "Resource": [
                "arn:aws:codedeploy:us-west-2:123456789012:deploymentgroup:*",
                "arn:aws:codebuild:us-east-1:123456789012:project/my-demo-project"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
                "s3:GetObject",
                "s3:DeletObject",
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": [
                "arn:aws:s3:::developer_bucket",
                "arn:aws:s3:::developer_bucket/*",
                "arn:aws:autoscling:us-east-2:123456789012:autoscalgrp"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": [
                        "public-read"
                    ],
                    "s3:prefix": [
                        "custom",
                        "other"
```

```
                    ]
                }
            }
        }
    ]
}
```

Comprendre les niveaux d'accès dans les résumés des politiques

AWS résumé des niveaux d'accès

Les récapitulatifs de politique comprennent un récapitulatif de niveau d'accès décrivant les autorisations d'action définies pour chaque service mentionné dans la politique. Pour en savoir plus sur les récapitulatifs de politiques, consultez Comprendre les autorisations accordées par une politique. Les récapitulatifs de niveau d'accès indiquent si les actions dans chaque niveau d'accès (`List`, `Read`, `Tagging`, `Write`, et `Permissions management`) ont des autorisations `Full` ou `Limited` définies dans la politique. Pour consulter la classification des niveaux d'accès attribuée à chaque action d'un service, consultez la section Actions, ressources et clés de condition pour les AWS services.

L'exemple suivant illustre l'accès fourni par une politique pour les services donnés. Pour consulter des exemples de documents de politique JSON complets et les récapitulatifs associés, consultez Exemples de récapitulatifs de la politique.

Service	Niveau d'accès	Cette politique fournit les autorisations suivantes :
IAM	Accès complet à	Accès à toutes les actions au sein du service IAM.
CloudWatch	Complet : List (Liste)	Accès à toutes les CloudWatch actions du niveau `List` d'accès, mais pas d'accès aux actions avec la classification `ReadWrite`, ou niveau `Permissions management` d'accès.
Data Pipeline	Limité : List (Liste), Read (Lire)	Accès à au moins une des AWS Data Pipeline actions du niveau d'`Read`accès `List` et, mais pas à toutes les actions, mais pas aux

Service	Niveau d'accès	Cette politique fournit les autorisations suivantes :
		`Permissions management` actions `Write` ou.
EC2	Complet : List (Liste), Read (Lire) Limité : Write (Écrire)	Accès à toutes les actions Amazon EC2 `List` et `Read`, et accès à au moins une action Amazon EC2 `Write`, mais à pas toutes, mais aucun accès aux actions ayant la classific ation de niveau d'accès `Permissions management` .
S3	Limité : Read (Lire), Write (Écrire), Permissions management (Gestion des autorisations)	Accès à au moins une action Amazon S3 `Read`, `Write` et `Permissions management` , mais pas toutes à la fois.
CodeDeploy	(empty)	Accès inconnu, car IAM ne reconnaît pas ce service.
API Gateway	Aucun	Aucun accès n'est défini dans la politique.
CodeBuild	 Aucune action n'est définie.	Aucun accès, car aucune action n'est définie pour le service. Pour comprendre et corriger ce problème, veuillez consulter the section called “Ma politique n'accorde pas les autorisations escomptées”.

Comme mentionné précédemment, l'accès complet indique que la politique autorise l'accès à toutes les actions contenues dans le service. Les politiques permettant l'accès à certaines, mais pas à l'ensemble des actions contenues dans un service sont regroupées en fonction de la classification de niveau d'accès. Cela est indiqué par un des regroupements de niveaux d'accès suivants :

- Complet : La politique fournit un accès complet à toutes les actions contenues dans la classification de niveau d'accès spécifiée.

- Limité : La politique fournit un accès à une ou plusieurs actions, mais pas à l'ensemble des actions contenues dans la classification de niveau d'accès spécifiée.
- Aucun : La politique ne fournit aucun accès.
- (vide) : IAM ne reconnaît pas ce service. Si le nom du service comprend une faute de frappe, la politique ne fournit aucun accès au service. Si le nom du service est correct, il se peut que le service ne prenne pas en charge les récapitulatifs de politique ou qu'il soit en mode aperçu. Dans ce cas, la politique peut accorder l'accès, mais celui-ci ne peut pas être affiché dans le récapitulatif de la politique. Pour demander la prise en charge du récapitulatif de politique d'un service disponible pour tous (GA), consultez Le service ne prend pas en charge les récapitulatifs de politique IAM.

Les résumés des niveaux d'accès qui incluent un accès limité (partiel) aux actions sont regroupés selon les classifications des niveaux d' AWS accès `ListRead`,, `TaggingWrite`, ou`Permissions management`.

AWS niveaux d'accès

AWS définit les classifications de niveaux d'accès suivantes pour les actions d'un service :

- List (Liste) : Autorisation de répertorier les ressources au sein du service afin de déterminer si un objet existe. Les actions associées à ce niveau d'accès peuvent répertorier les objets mais ne peuvent pas voir le contenu d'une ressource. Par exemple, l'action `ListBucket` Amazon S3 possède le niveau d'accès List (Liste).
- Read (Lire) : Autorisation de lire le contenu et les attributs de ressources dans le service, mais pas de les modifier. Par exemple, les actions Amazon S3 `GetObject` et `GetBucketLocation` possèdent le niveau d'accès Read (Lecture).
- Balisage : Autorisation d'effectuer des actions qui modifient uniquement l'état des balises de ressource. Par exemple, les actions IAM `TagRole` et `UntagRole` ont le niveau d'accès Tagging (Balisage), car elles autorisent uniquement le balisage ou l'annulation du balisage d'un rôle. Cependant, l'action `CreateRole` autorise le balisage d'une ressource de rôle lorsque vous créez ce rôle. Étant donné que l'action n'ajoute pas uniquement une balise, elle possède le niveau d'accès `Write`.
- Write (Écrire) : autorisation de créer, supprimer ou modifier des ressources du service. Par exemple, les actions Amazon S3 `CreateBucket` `DeleteBucket` et `PutObject` possèdent le niveau d'accès Write (Écriture). Les actions `Write` peuvent également autoriser la modification

d'une balise de ressource. Toutefois, une action qui autorise uniquement les modifications des balises possède le niveau d'accès `Tagging`.

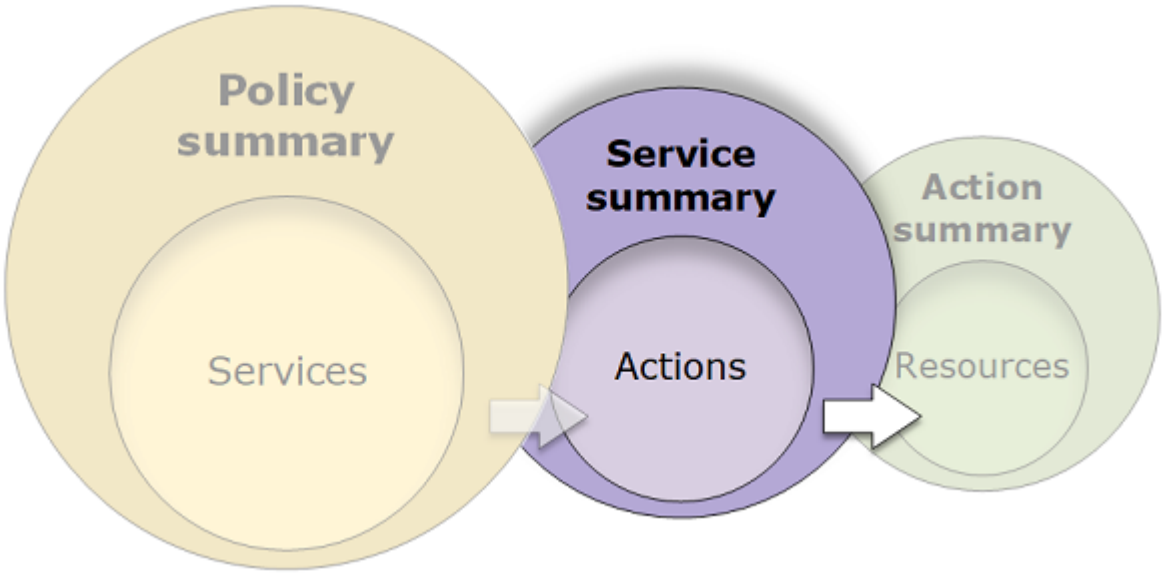
- Permissions management (Gestion des autorisations) : Autorisation d'octroyer ou de modifier des autorisations de ressource dans le service. Par exemple, la plupart des IAM et AWS Organizations des actions, ainsi que des actions telles que les actions Amazon S3, `DeleteBucketPolicy` ont un niveau `PutBucketPolicy` d'accès à la gestion des autorisations.

> **Conseil**
>
> Pour améliorer votre sécurité Compte AWS, limitez ou surveillez régulièrement les politiques qui incluent la classification des niveaux d'accès à la gestion des autorisations.

Pour consulter la classification des niveaux d'accès pour toutes les actions d'un service, voir Actions, ressources et clés de condition pour les AWS services.

Récapitulatif du service (liste des actions)

Les politiques sont résumées dans trois tables : récapitulatif de la politique, récapitulatif du service et récapitulatif de l'action. La table du récapitulatif du service inclut une liste des actions et des récapitulatifs des autorisations définies par la politique pour le service choisi.



Vous pouvez consulter un récapitulatif du service pour chaque service répertorié dans le récapitulatif de la politique qui accorde les autorisations. La table est regroupée en sections Uncategorized actions (Actions non catégorisées), Uncategorized resource types (Types de ressources non

catégorisées) et de niveau d'accès. Si la politique comprend une action qu'IAM ne reconnaît pas, l'action est incluse dans la section Uncategorized actions (Actions non catégorisées) de la table. Si IAM reconnaît l'action, elle est alors incluse dans l'une des sections de niveau d'accès (liste, lecture, écriture et gestion des autorisations) du tableau. Pour consulter la classification des niveaux d'accès attribuée à chaque action d'un service, consultez la section Actions, ressources et clés de condition pour les AWS services.

Affichage des récapitulatifs du service

Vous pouvez afficher le résumé des services pour les politiques gérées sur la page Politiques.

Pour afficher le récapitulatif du service d'une politique gérée

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de la stratégie à afficher.
4. Sur la page Détails de la politique, affichez l'onglet Autorisations pour consulter le résumé de la politique.
5. Dans la liste des services du récapitulatif de la politique, choisissez le nom du service à afficher.

Pour afficher le récapitulatif du service d'une politique attachée à un utilisateur

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez utilisateurs.
3. Dans la liste des utilisateurs, choisissez le nom de l'utilisateur dont vous souhaitez afficher la politique.
4. Sur la page Récapitulatif de l'utilisateur, affichez l'onglet Autorisations pour afficher la liste des stratégies attachées à l'utilisateur directement ou via un groupe.
5. Dans la table des politiques de l'utilisateur, choisissez le nom de la politique que vous voulez afficher.

   Si vous vous trouvez sur la page Utilisateurs et vous choisissez d'afficher le résumé des services pour une politique attachée à cet utilisateur, vous êtes redirigé vers la page Politiques. Vous pouvez afficher les résumés des services uniquement sur la page Politiques.

6. Choisissez Résumé. Dans la liste des services du récapitulatif de la politique, choisissez le nom du service à afficher.

> **Note**
>
> Si la politique que vous sélectionnez est une politique en ligne attachée directement à l'utilisateur, le tableau récapitulatif du service s'affiche. Si la politique est une politique en ligne attachée via un groupe, vous êtes dirigé vers le document de politique JSON de ce groupe. Si la stratégie est une stratégie gérée, vous êtes dirigé vers le récapitulatif du service de cette stratégie sur la page Stratégies.

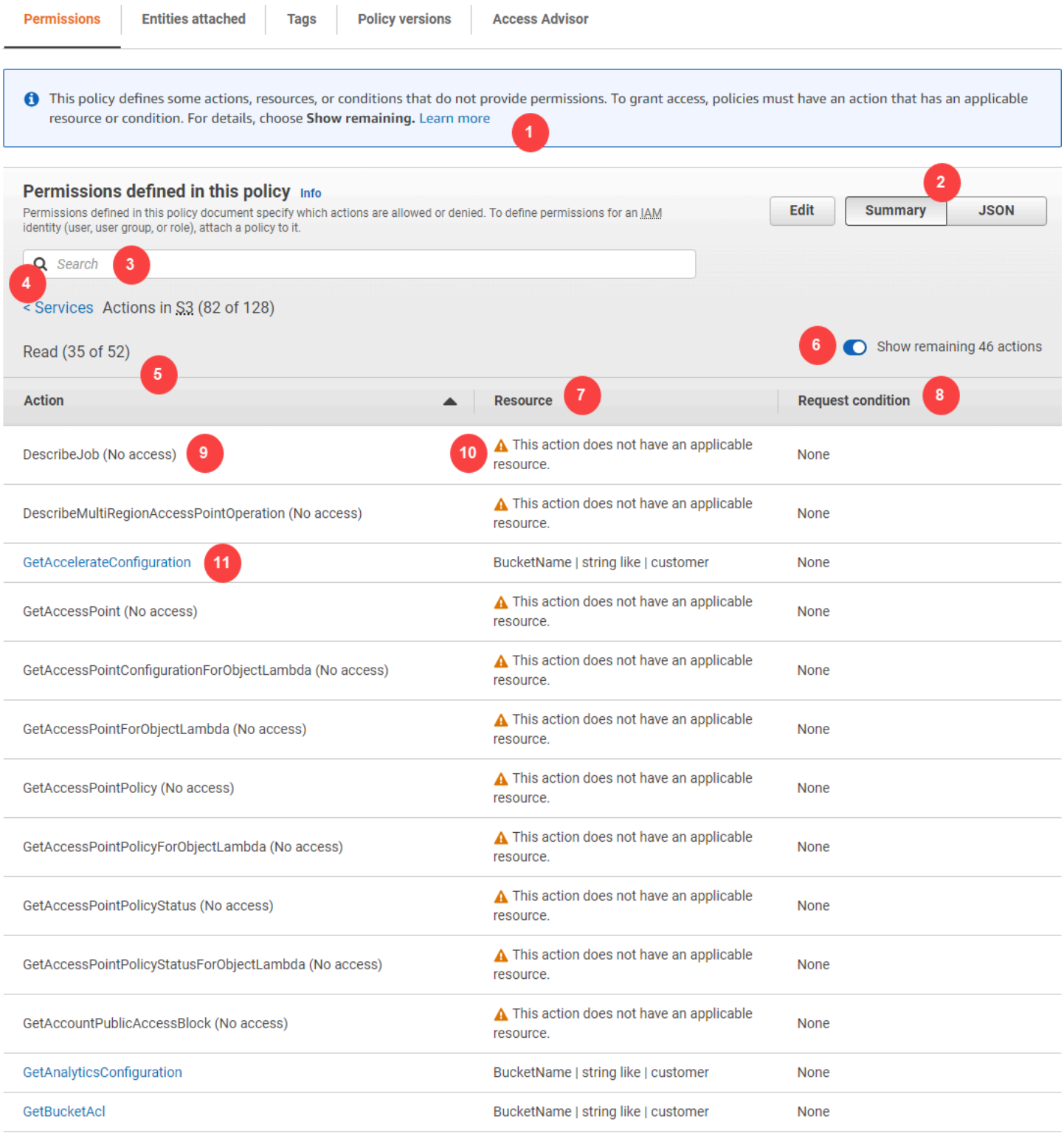
Pour afficher le récapitulatif du service d'une politique attachée à un rôle

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Rôles dans le panneau de navigation.
3. Dans la liste des rôles, choisissez le nom du rôle dont vous souhaitez afficher la politique.
4. Sur la page Récapitulatif du rôle, affichez l'onglet Autorisations pour afficher la liste des stratégies attachées au rôle.
5. Dans la table des politiques pour le rôle, choisissez le nom de la politique que vous voulez afficher.

   Si vous vous trouvez sur la page Rôles et vous choisissez d'afficher le résumé des services pour une politique attachée à cet utilisateur, vous êtes redirigé vers la page Politiques. Vous pouvez afficher les résumés des services uniquement sur la page Politiques.
6. Dans la liste des services du récapitulatif de la politique, choisissez le nom du service à afficher.

Présentation des éléments d'un récapitulatif du service

L'exemple suivant représente le résumé des services pour les actions Amazon S3 qui sont autorisées à partir d'un résumé de la politique. Les actions de ce service sont groupées par niveau d'accès. Par exemple, 35 actions Lire sont définies sur un total de 52 actions Lire disponibles pour le service.



La page de récapitulatif du service pour une politique gérée inclut les informations suivantes :

1. Si la politique n'accorde pas les autorisations pour toutes les actions, ressources et conditions définies pour le service dans la politique, un avertissement ou une bannière d'erreur s'affiche

dans la partie supérieure de la page. Puis le récapitulatif du service inclut des détails relatifs au problème. Pour savoir comment les récapitulatifs de politique vous aident à comprendre les autorisations que votre politique vous accorde et à résoudre les problèmes les concernant, consultez the section called “Ma politique n'accorde pas les autorisations escomptées”.

2. Choisissez JSON pour consulter des détails supplémentaires sur la politique. Cela peut vous servir pour afficher l'ensemble des conditions appliquées aux actions. (Si vous consultez le récapitulatif du service pour une politique en ligne attachée directement à un utilisateur, vous devez fermer la boîte de dialogue du récapitulatif du service et revenir au récapitulatif de la politique pour accéder au document de politique JSON.)
3. Pour consulter le résumé d'une action spécifique, tapez des mots-clés dans la zone de Recherche afin de réduire la liste des actions disponibles.
4. En regard de la flèche de retour Services apparaît le nom du service (dans ce cas S3). Le résumé des services pour ce service inclut la liste des actions autorisées ou refusées définies dans la politique. Si le service apparaît sous (Refus explicite) dans l'onglet Autorisations, alors les actions énumérées dans la table récapitulative des services sont explicitement refusées. Si le service apparaît sous Autoriser dans l'onglet Autorisations, alors les actions énumérées dans la table récapitulative des services sont autorisées.
5. Action : cette colonne énumère les actions définies dans la politique et fournit les ressources et les conditions pour chaque action. Si la politique accorde ou refuse des autorisations à l'action, alors le nom de l'action renvoie à la table récapitulative de l'action. La table regroupe ces actions dans au moins une ou jusqu'à cinq sections au maximum, en fonction du niveau d'accès autorisé ou refusé par la politique. Les sections sont Liste, Lire, Écrire, Gestion des autorisations et Balisage. Le compte indique le nombre d'actions reconnues qui fournissent des autorisations pour chaque niveau d'accès. Le total correspond au nombre d'actions connues pour le service. Dans cet exemple, 35 actions fournissent des autorisations sur un total de 52 actions connues Amazon S3 Lire. Pour consulter la classification des niveaux d'accès attribuée à chaque action d'un service, consultez la section Actions, ressources et clés de condition pour les AWS services.
6. Afficher les actions restantes : activez ce bouton pour développer ou masquer la table afin d'inclure des actions connues mais qui ne fournissent pas d'autorisations pour ce service. L'activation du bouton permet également d'afficher des avertissements pour tous les éléments qui ne fournissent pas d'autorisations.
7. Resource (Ressource) : cette colonne présente les ressources que la politique définit pour le service. IAM ne vérifie pas si la ressource s'applique à chaque action. Dans cet exemple, les actions du service Amazon S3 sont autorisées uniquement sur la ressource du compartiment Amazon S3 `developer_bucket`. En fonction des informations que le service fournit à IAM,

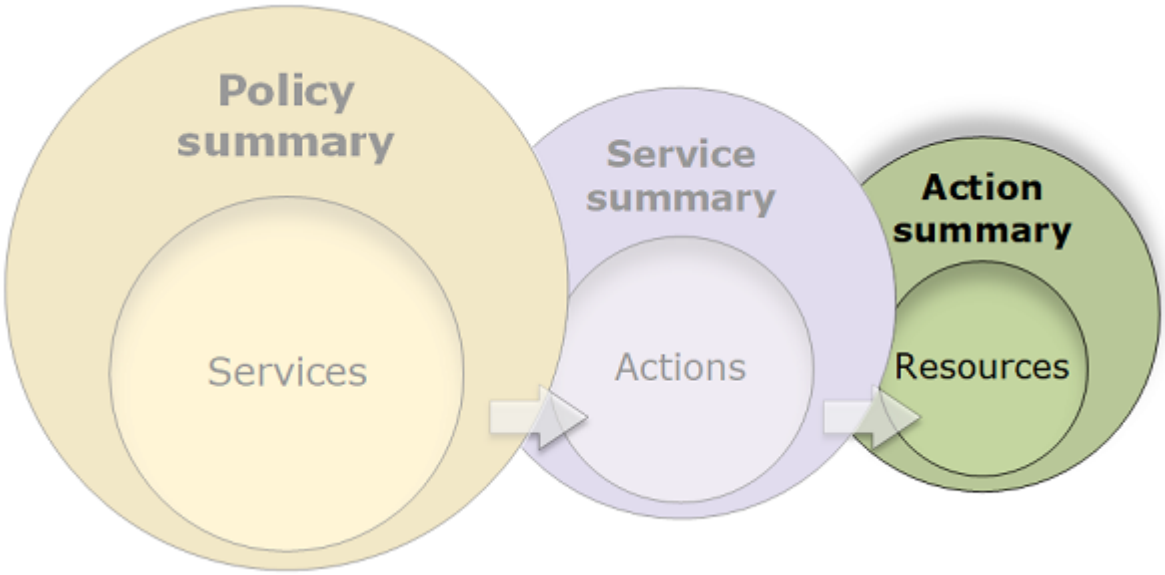
vous verrez s'afficher un ARN, tel que `arn:aws:s3:::developer_bucket/*`, ou le type de ressource défini, tel que `BucketName = developer_bucket`.

> **Note**
>
> Cette colonne peut contenir une ressource provenant d'un service différent. Si l'instruction de politique qui inclut la ressource ne contient pas d'actions et de ressources provenant d'un même service, cela signifie que votre politique contient des ressources non appariées. IAM ne vous avertit pas que des ressources ne sont pas appariées lorsque vous créez une politique ou que vous affichez une politique dans le récapitulatif du service. IAM n'indique pas non plus si l'action s'applique aux ressources, uniquement si le service correspond. Si cette colonne contient une ressource non appariée, recherchez d'éventuelles erreurs dans votre politique. Afin de mieux comprendre vos politiques, vous devez toujours les tester avec le simulateur de politique.

8. Request condition (Demander une condition) : cette colonne indique si les services ou actions associés aux ressources font l'objet de conditions. Pour en savoir plus sur ces conditions, choisissez JSON pour examiner le document de politique JSON.
9. (No access) (Pas d'accès) : cette politique inclut une action qui ne fournit pas d'autorisations.
10. Resource warning (Avertissement concernant les ressources) : pour les actions dont les ressources ne fournissent pas d'autorisations complètes, l'un des avertissements suivants s'affiche :
    - Cette action ne prend pas en charge les autorisations au niveau des ressources. Cet élément exige un caractère générique (*) pour la ressource. – Cela signifie que la politique inclut des autorisations au niveau des ressources, mais qu'elle doit inclure `"Resource": ["*"]` pour fournir des autorisations pour cette action.
    - Cette action n'a pas de ressource applicable. – Cela signifie que l'action est incluse dans la politique sans ressource prise en charge.
    - Cette action n'a pas de ressource et de condition applicable. – Cela signifie que l'action est incluse dans la politique sans ressource prise en charge, ni condition prise en charge. Dans ce cas, il y a également une condition incluse dans la politique pour ce service, mais il n'existe pas de conditions qui s'appliquent à cette action.
11. Les actions qui fournissent des autorisations incluent un lien vers le récapitulatif de l'action.

Récapitulatif de l'action (liste des ressources)

Les politiques sont résumées dans trois tables : récapitulatif de la politique, récapitulatif du service et récapitulatif de l'action. La table récapitulative de l'action inclut une liste des ressources et des conditions associées qui s'appliquent à l'action choisie.



Pour afficher un récapitulatif de l'action pour chaque action qui accorde des autorisations, choisissez le lien dans le récapitulatif du service. La table récapitulative de l'action inclut des détails concernant la ressource, notamment sa Région et son Compte. Vous pouvez également consulter les conditions qui s'appliquent à chaque ressource. Cela affiche les conditions s'appliquant à certaines ressources mais pas à d'autres.

Affichage des récapitulatifs des actions

Vous pouvez afficher le résumé des actions pour les politiques gérées, toute politique attachée à un utilisateur et toute politique attachée à un rôle sur la page Politiques.

Pour afficher le récapitulatif de l'action d'une politique gérée

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Politiques.
3. Dans la liste des stratégies, choisissez le nom de la stratégie à afficher.
4. Sur la page Détails de la politique, affichez l'onglet Autorisations pour consulter le résumé de la politique.

5. Dans la liste des services du récapitulatif de la politique, choisissez le nom du service à afficher.
6. Dans la liste des actions du récapitulatif du service, choisissez le nom de l'action à afficher.

Pour afficher le récapitulatif de l'action d'une politique attachée à un utilisateur

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Utilisateurs dans le panneau de navigation.
3. Dans la liste des utilisateurs, choisissez le nom de l'utilisateur dont vous souhaitez afficher la politique.
4. Sur la page Récapitulatif de l'utilisateur, affichez l'onglet Autorisations pour afficher la liste des stratégies attachées à l'utilisateur directement ou via un groupe.
5. Dans la table des politiques de l'utilisateur, choisissez le nom de la politique que vous voulez afficher.

   Si vous vous trouvez sur la page Utilisateurs et vous choisissez d'afficher le résumé des services pour une politique attachée à cet utilisateur, vous êtes redirigé vers la page Politiques. Vous pouvez afficher les résumés des services uniquement sur la page Politiques.
6. Dans la liste des services du récapitulatif de la politique, choisissez le nom du service à afficher.

   > **Note**
   >
   > Si la politique que vous sélectionnez est une politique en ligne attachée directement à l'utilisateur, le tableau récapitulatif du service s'affiche. Si la politique est une politique en ligne attachée via un groupe, vous êtes dirigé vers le document de politique JSON de ce groupe. Si la stratégie est une stratégie gérée, vous êtes dirigé vers le récapitulatif du service de cette stratégie sur la page Stratégies.
7. Dans la liste des actions du récapitulatif du service, choisissez le nom de l'action à afficher.

Pour afficher le récapitulatif d'action d'une politique attachée à un rôle

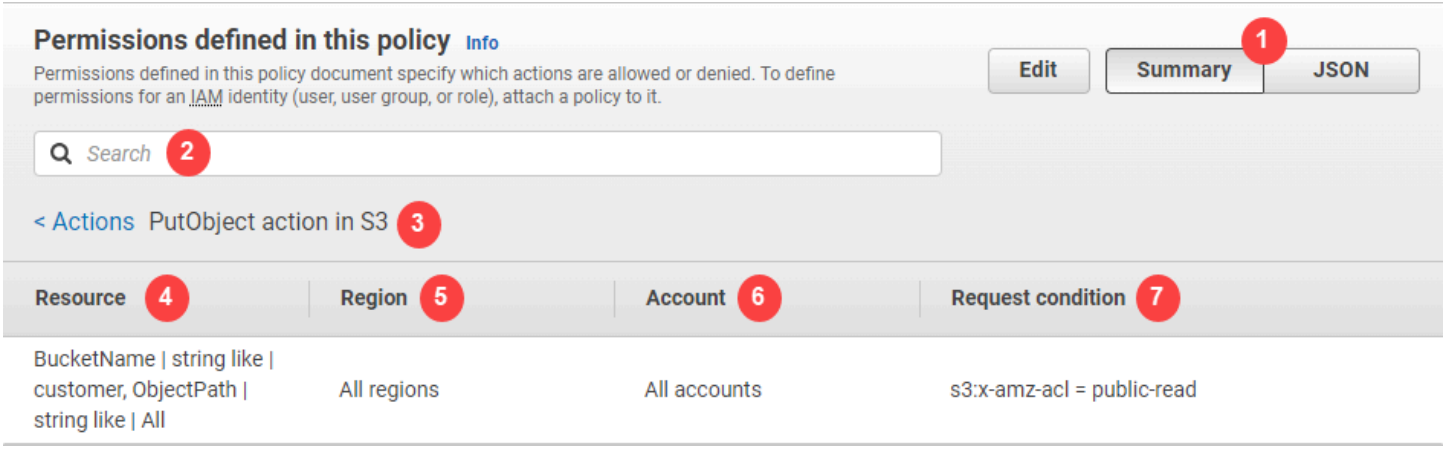
1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Roles (Rôles).
3. Dans la liste des rôles, choisissez le nom du rôle dont vous souhaitez afficher la politique.

4. Sur la page Récapitulatif du rôle, affichez l'onglet Autorisations pour afficher la liste des stratégies attachées au rôle.
5. Dans la table des politiques pour le rôle, choisissez le nom de la politique que vous voulez afficher.

   Si vous vous trouvez sur la page Rôles et vous choisissez d'afficher le résumé des services pour une politique attachée à cet utilisateur, vous êtes redirigé vers la page Politiques. Vous pouvez afficher les résumés des services uniquement sur la page Politiques.
6. Dans la liste des services du récapitulatif de la politique, choisissez le nom du service à afficher.
7. Dans la liste des actions du récapitulatif du service, choisissez le nom de l'action à afficher.

Présentation des éléments d'un récapitulatif d'action

L'exemple ci-dessous illustre le récapitulatif de l'action `PutObject` (écriture) du récapitulatif de service Amazon S3 (consultez Récapitulatif du service (liste des actions)). Pour cette action, la politique définit plusieurs conditions sur une seule ressource.



La page de récapitulatif de l'action comprend les informations suivantes :

1. Choisissez JSON pour consulter des détails supplémentaires sur la politique, tels que les conditions multiples appliquées aux actions. (Si vous affichez le résumé d'action d'une politique en ligne qui est directement attachée à un utilisateur, les étapes diffèrent. Pour accéder au document de politique JSON dans ce cas, vous devez fermer la boîte de dialogue de résumé d'action et revenir au résumé de politique.)
2. Pour afficher le résumé d'une ressource spécifique, tapez des mots-clés dans la zone de Recherche afin de réduire la liste des ressources disponibles.

3. À côté de la flèche de retour des actions apparaissent le nom du service et de l'action au format `action name action in service` (dans ce cas, PutObjectaction dans S3). Le récapitulatif de l'action pour ce service inclut la liste des ressources autorisées définies dans la politique.
4. Resource (Ressource) : cette colonne présente les ressources que la politique définit pour le service choisi. Dans cet exemple, l'PutObjectaction est autorisée sur tous les chemins d'objets, mais uniquement sur la ressource du compartiment `developer_bucket` Amazon S3. En fonction des informations que le service fournit à IAM, vous verrez s'afficher un ARN, tel que `arn:aws:s3:::developer_bucket/*`, ou le type de ressource défini, tel que `BucketName = developer_bucket, ObjectPath = All`.
5. Region (Région) : cette colonne présente la région dans laquelle la ressource est définie. Il est possible de définir des ressources pour toutes les régions ou pour une seule région. Les ressources ne peuvent exister que dans une région spécifique.
   - Toutes les régions : les actions associées à la ressource s'appliquent à toutes les régions. Dans cet exemple, l'action appartient à un service mondial, Amazon S3. Les actions qui appartiennent aux services mondiaux s'appliquent à toutes les régions.
   - Region text (Texte de région) : les actions associées à la ressource s'appliquent à une région. Par exemple, une politique peut spécifier la région `us-east-2` pour une ressource.
6. Account (Compte) : cette colonne indique si les services ou actions associés à la ressource s'appliquent à un compte spécifique. Les ressources peuvent exister dans tous les comptes ou dans un seul. Celles-ci ne peuvent pas exister dans un compte spécifique.
   - All accounts (Tous les comptes) : les actions associées à la ressource s'appliquent à tous les comptes. Dans cet exemple, l'action appartient à un service mondial, Amazon S3. Les actions qui appartiennent aux services mondiaux s'appliquent à tous les comptes.
   - Ce compte : les actions associées à la ressource s'appliquent uniquement au compte auquel vous êtes connecté.
   - Account number (Numéro du compte) : les actions associées à la ressource s'appliquent uniquement à un compte (un compte auquel vous n'êtes pas connecté). Par exemple, si une politique précise le compte `123456789012` pour une ressource, le numéro de compte apparaît dans le récapitulatif de la politique.
7. Request condition (Demander une condition) : cette colonne indique si les actions qui sont associées aux ressources font l'objet de conditions. Cet exemple inclut la condition `s3:x-amz-acl = public-read`. Pour en savoir plus sur ces conditions, choisissez JSON pour examiner le document de politique JSON.

Exemples de récapitulatifs de la politique

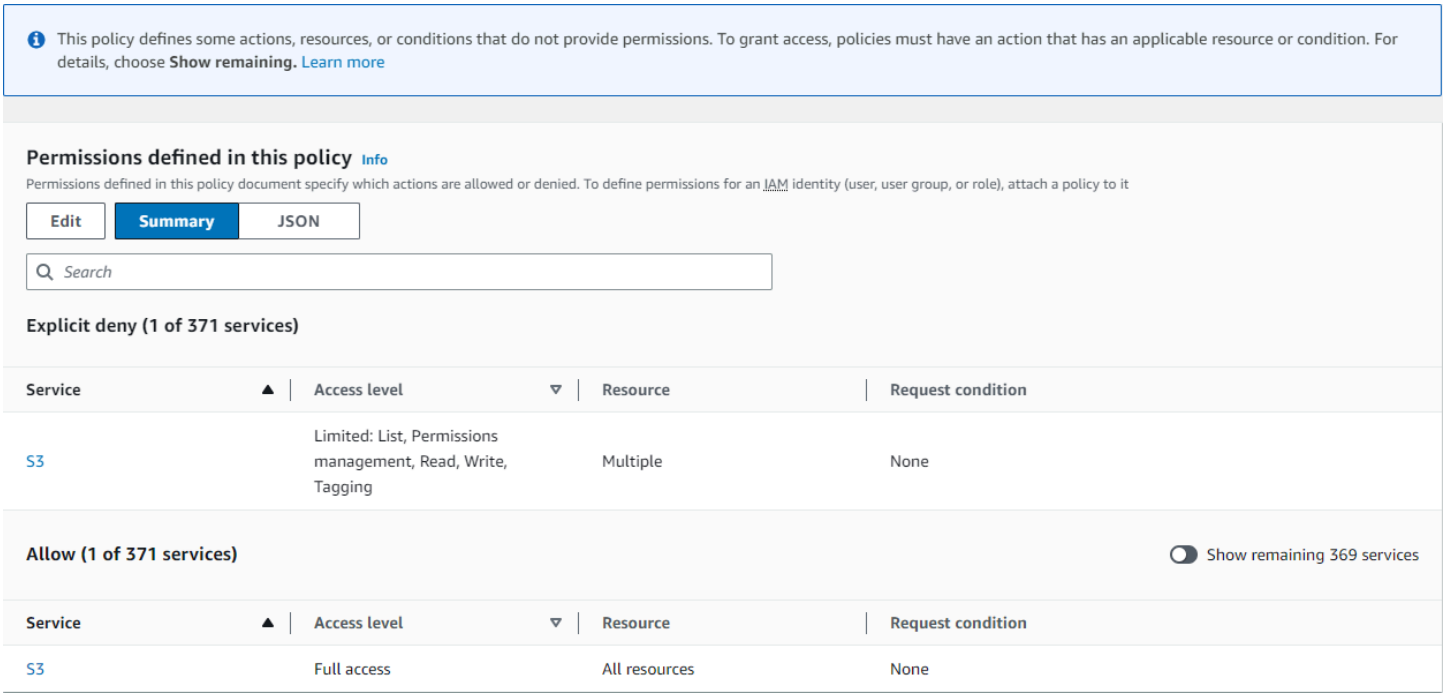
Les exemples suivants incluent des politiques JSON et leurs récapitulatifs de la politique associés, des récapitulatifs du service, ainsi que des récapitulatifs de l'action pour vous aider à comprendre les autorisations accordées par le biais d'une politique.

Politique 1 : DenyCustomerBucket

Cette politique illustre une autorisation et un refus pour le même service.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccess",
            "Effect": "Allow",
            "Action": ["s3:*"],
            "Resource": ["*"]
        },
        {
            "Sid": "DenyCustomerBucket",
            "Action": ["s3:*"],
            "Effect": "Deny",
            "Resource": ["arn:aws:s3:::customer", "arn:aws:s3:::customer/*" ]
        }
    ]
}
```

DenyCustomerBucketRésumé de la politique :



DenyCustomerBucket Résumé du service S3 (refus explicite) :

< Services Actions in S3 (82 of 130) Show remaining 48 actions

Read (35 of 53)

Action	Resource	Request condition
GetAccelerateConfiguration	BucketName| string like |customer	None
GetAnalyticsConfiguration	BucketName| string like |customer	None
GetBucketAcl	BucketName| string like |customer	None
GetBucketCORS	BucketName| string like |customer	None
GetBucketLocation	BucketName| string like |customer	None
GetBucketLogging	BucketName| string like |customer	None
GetBucketNotification	BucketName| string like |customer	None
GetBucketObjectLockConfiguration	BucketName| string like |customer	None
GetBucketOwnershipControls	BucketName| string like |customer	None
GetBucketPolicy	BucketName| string like |customer	None
GetBucketPolicyStatus	BucketName| string like |customer	None
GetBucketPublicAccessBlock	BucketName| string like |customer	None
GetBucketRequestPayment	BucketName| string like |customer	None
GetBucketTagging	BucketName| string like |customer	None
GetBucketVersioning	BucketName| string like |customer	None
GetBucketWebsite	BucketName| string like |customer	None

GetObject (Lire) Résumé des actions :

< Actions GetObject action in S3

Resource	Region	Account	Request condition
BucketName| string like |customer, ObjectPath| string like |All	-	All accounts	None

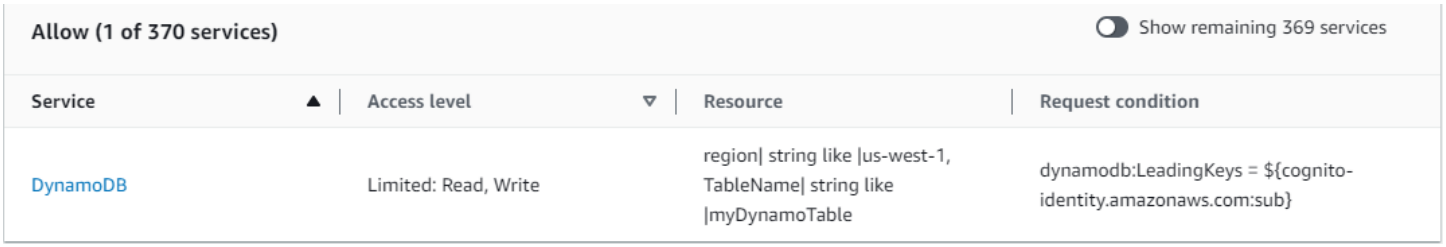
Politique 2 : DynamoDbRowCognito ID

Cette politique offre un accès de niveau ligne à Amazon DynamoDB en fonction de l'ID Amazon Cognito de l'utilisateur.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": [
                "arn:aws:dynamodb:us-west-1:123456789012:table/myDynamoTable"
            ],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": [
                        "${cognito-identity.amazonaws.com:sub}"
                    ]
                }
            }
        }
    ]
}
```

DynamoDbRowCognitoRésumé de la politique d'identification :



DynamoDbRowCognitoRésumé du service ID DynamoDB (Allow) :

< Services Actions in DynamoDB (4 of 65) Show remaining 61 actions

Read (1 of 26)

Action	Resource	Request condition
GetItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

Write (3 of 33)

Action	Resource	Request condition
DeleteItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
PutItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
UpdateItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

GetItem Résumé des actions (liste) :

< Actions GetItem action in DynamoDB

Resource	Region	Account	Request condition
region| string like |us-west-1, TableName| string like |myDynamoTable	us-west-1	123456789012	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

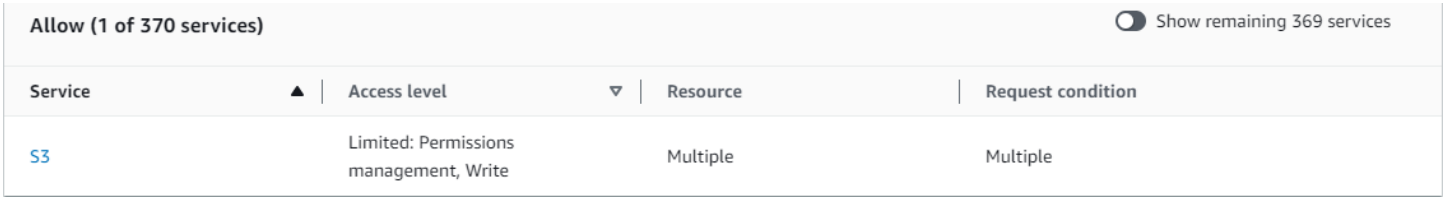
Politique 3 : MultipleResourceCondition

Cette politique comprend plusieurs ressources et conditions.

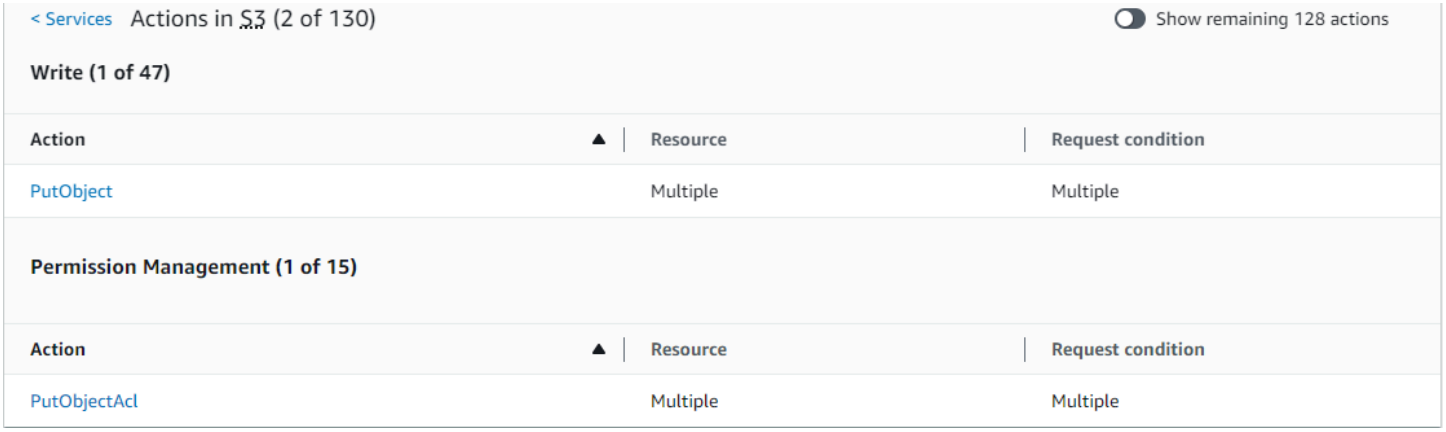
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Apple_bucket/*"],
            "Condition": {"StringEquals": {"s3:x-amz-acl": ["public-read"]}}
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Orange_bucket/*"],
            "Condition": {"StringEquals": {
                "s3:x-amz-acl": ["custom"],
                "s3:x-amz-grant-full-control": ["1234"]
            }}
        }
    ]
}
```

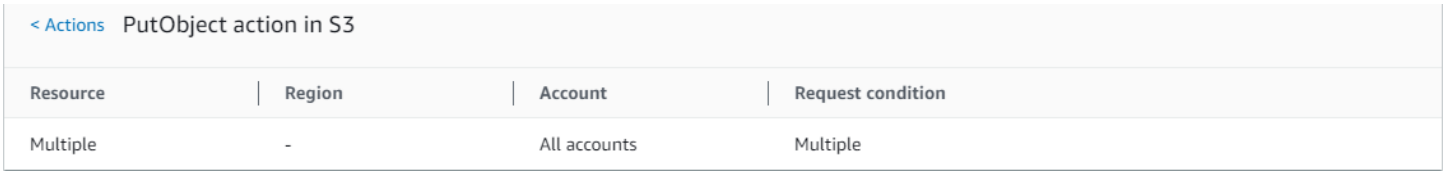
MultipleResourceConditionRésumé de la politique :



MultipleResourceCondition Résumé du service S3 (Autoriser) :



PutObject (Écrire) Résumé des actions :

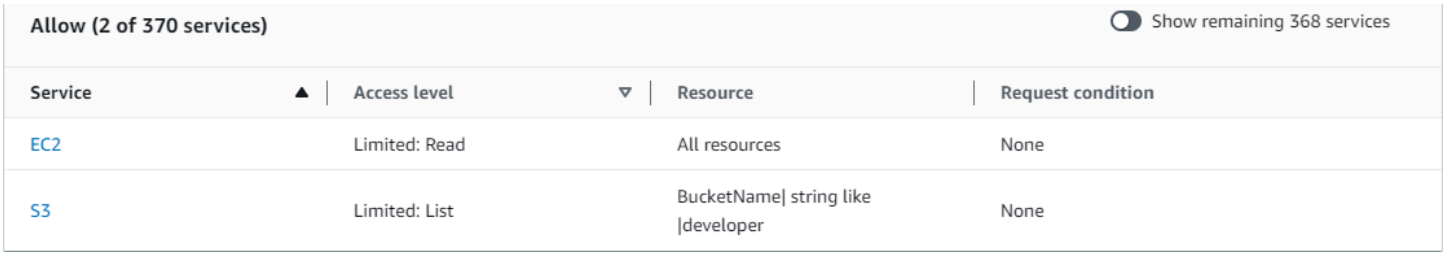


Politique 4 : EC2_troubleshoot

La politique suivante permet aux utilisateurs de récupérer une capture d'écran d'une instance Amazon EC2 en cours d'exécution, ce qui facilite la résolution des problèmes liés à EC2. Cette politique permet également de consulter des informations sur les éléments du compartiment du développeur Amazon S3.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshot"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::developer"
            ]
        }
    ]
}
```

EC2_Troubleshoot Récapitulatif de la politique :



Allow (2 of 370 services) — Show remaining 368 services

Service	Access level	Resource	Request condition
EC2	Limited: Read	All resources	None
S3	Limited: List	BucketName| string like |developer	None

EC2_Troubleshoot S3 (Allow) Récapitulatif du service :

< Services Actions in S3 (1 of 130) Show remaining 129 actions

List (1 of 7)

Action	Resource	Request condition
ListBucket	BucketName| string like |developer	None

ListBucket Résumé des actions (liste) :

< Actions ListBucket action in S3

Resource	Region	Account	Request condition
BucketName| string like |developer	-	All accounts	None

Politique 5 : CodeBuild _ CodeCommit _ CodeDeploy

Cette politique donne accès à CodeBuild CodeCommit des CodeDeploy ressources spécifiques. Ces ressources apparaissent uniquement avec le service correspondant car elles sont propres à chaque service. Si vous incluez une ressource qui ne correspond à aucun service de l'élément `Action`, elle apparaît dans tous les récapitulatifs de l'action.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1487980617000",
            "Effect": "Allow",
            "Action": [
                "codebuild:*",
                "codecommit:*",
                "codedeploy:*"
            ],
            "Resource": [
                "arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project",
                "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo",
                "arn:aws:codedeploy:us-east-2:123456789012:application:WordPress_App",
                "arn:aws:codedeploy:us-east-2:123456789012:instance/AssetTag*"
            ]
        }
    ]
}
```

CodeBuild_ CodeCommit _ Résumé CodeDeploy de la politique :

Allow (3 of 370 services) Show remaining 367 services

Service	Access level	Resource	Request condition
CodeBuild	Full: Permissions management Limited: List, Read, Write	region| string like |us-east-2	None
CodeCommit	Full: Tagging Limited: List, Read, Write	ResourceSpecifier| string like |MyDemoRepo, region| string like |us-east-2	None
CodeDeploy	Full: Tagging Limited: List, Read, Write	Multiple	None

CodeBuild_ CodeCommit _ CodeDeploy CodeBuild (Autoriser) Résumé du service :

< Services Actions in CodeBuild (24 of 53) Show remaining 29 actions

Read (4 of 9)

Action	Resource	Request condition
BatchGetBuildBatches	region| string like |us-east-2	None
BatchGetBuilds	region| string like |us-east-2	None
BatchGetProjects	region| string like |us-east-2	None
GetResourcePolicy	region| string like |us-east-2	None

Write (16 of 28)

Action	Resource	Request condition
BatchDeleteBuilds	region| string like |us-east-2	None
CreateProject	region| string like |us-east-2	None
CreateWebhook	region| string like |us-east-2	None
DeleteBuildBatch	region| string like |us-east-2	None
DeleteProject	region| string like |us-east-2	None
DeleteWebhook	region| string like |us-east-2	None
InvalidateProjectCache	region| string like |us-east-2	None
RetryBuild	region| string like |us-east-2	None
RetryBuildBatch	region| string like |us-east-2	None
StartBuild	region| string like |us-east-2	None
StartBuildBatch	region| string like |us-east-2	None
StopBuild	region| string like |us-east-2	None
StopBuildBatch	region| string like |us-east-2	None
UpdateProject	region| string like |us-east-2	None
UpdateProjectVisibility	region| string like |us-east-2	None
UpdateWebhook	region| string like |us-east-2	None

List (2 of 14)

CodeBuild_ CodeCommit _ CodeDeploy StartBuild (Écrire) Résumé des actions :

< Actions StartBuild action in CodeBuild

Resource	Region	Account	Request condition
region| string like |us-east-2	us-east-2	123456789012	None

Autorisations requises pour accéder aux autres ressources IAM

Les ressources sont des objets au sein d'un service. Les ressources IAM incluent des groupes, des utilisateurs, des rôles et des politiques. Si vous êtes connecté avec des informations d'identification Utilisateur racine d'un compte AWS , aucune restriction ne vous est imposée en termes d'administration des informations d'identification IAM ou des ressources IAM. Cependant, les utilisateurs IAM doivent explicitement accorder des autorisations pour gérer les informations d'identification ou des ressources IAM. Pour ce faire, vous pouvez attacher une politique basée sur les identités à l'utilisateur.

> **Note**
>
> Dans la AWS documentation, lorsque nous faisons référence à une politique IAM sans mentionner aucune des catégories spécifiques, nous entendons une politique basée sur l'identité et gérée par le client. Pour de plus amples informations sur les catégories de politique, veuillez consulter the section called “Politiques et autorisations”.

Autorisations pour la gestion des identités IAM

Les autorisations qui sont requises pour administrer des groupes, des utilisateurs, des rôles et des informations d'identification IAM correspondent aux actions d'API de la tâche. Par exemple, pour créer des utilisateurs IAM, vous devez disposer de l'autorisation `iam:CreateUser` qui comporte la commande API correspondante : `CreateUser`. Pour autoriser un utilisateur IAM à créer d'autres utilisateurs IAM, vous pouvez attacher une politique IAM comme celle qui suit à cet utilisateur :

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

Dans une politique, la valeur de l'élément `Resource` dépend de l'action et des ressources que cette action peut affecter. Dans l'exemple précédent, la politique autorise un utilisateur à créer un autre utilisateur (* est un caractère générique qui correspond à toutes les chaînes). Par opposition, une

politique qui autorise des utilisateurs à modifier uniquement leurs propres clés d'accès (actions API `CreateAccessKey` et `UpdateAccessKey`) comporte généralement un élément `Resource`. Dans ce cas, l'ARN inclut une variable (`${aws:username}`) qui se résout par le nom de l'utilisateur actuel, comme dans l'exemple suivant :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListUsersForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "arn:aws:iam::*:*"
        },
        {
            "Sid": "ViewAndUpdateAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:UpdateAccessKey",
                "iam:CreateAccessKey",
                "iam:ListAccessKeys"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Dans l'exemple précédent, `${aws:username}` est une variable qui renvoie au nom utilisateur de l'utilisateur actuel. Pour de plus amples informations sur les variables de politique, veuillez consulter Éléments des politiques IAM : variables et balises.

L'utilisation d'un caractère générique (*) dans le nom d'action facilite souvent l'octroi d'autorisations pour toutes les actions associées à une tâche spécifique. Par exemple, pour autoriser des utilisateurs à exécuter n'importe quelle action IAM, vous pouvez utiliser `iam:*` pour l'action. Pour autoriser les utilisateurs à effectuer n'importe quelle action associée uniquement aux clés d'accès, vous pouvez utiliser `iam:*AccessKey*` dans l'élément `Action` d'une instruction de politique. Cela permet d'accorder à l'utilisateur l'autorisation d'effectuer les actions `CreateAccessKey`, `DeleteAccessKey`, `GetAccessKeyLastUsed`, `ListAccessKeys` et `UpdateAccessKey`. (Si une action dont le nom est « AccessKey » est ajoutée à IAM dans le futur, l'utilisation `iam:*AccessKey*` de l'`Action`élément donnera également à l'utilisateur l'autorisation d'effectuer cette nouvelle

action.) L'exemple suivant montre une politique qui permet aux utilisateurs d'effectuer toutes les actions relatives à leurs propres clés d'accès (*account-id*remplacez-les par votre Compte AWS identifiant) :

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/${aws:username}"
  }
}
```

Certaines tâches, comme la suppression d'un groupe, impliquent plusieurs actions : vous devez d'abord supprimer les utilisateurs du groupe, puis détacher ou supprimer les stratégie du groupe et enfin supprimer réellement le groupe. Si vous souhaitez qu'un utilisateur supprime un groupe, vous devez vous assurer d'accorder à celui-ci les autorisations nécessaires pour exécuter toutes les actions associées.

Autorisations pour utiliser AWS Management Console

Les exemples précédents illustrent des politiques autorisant un utilisateur à exécuter les actions avec l'interface AWS CLI ou les kits SDK AWS.

À mesure que les utilisateurs se servent de la console, celle-ci envoie des demandes à IAM pour répertorier les groupes, les utilisateurs, les rôles et les politiques, et pour obtenir les politiques associées à un groupe, un utilisateur ou un rôle. La console émet également des demandes pour obtenir Compte AWS des informations et des informations sur le principal. Le principal est l'utilisateur qui effectue les demandes sur la console.

En général, pour effectuer une action, vous devez posséder uniquement l'action correspondante dans une politique. Pour pouvoir créer un utilisateur, vous devez disposer d'une autorisation pour appeler l'action `CreateUser`. Souvent, lorsque vous utilisez la console pour effectuer une action, vous devez avoir les autorisations pour afficher, répertorier, obtenir, ou encore afficher les ressources sur la console. Ceci est nécessaire pour vous permettre de naviguer au sein de la console pour effectuer l'action spécifiée. Par exemple, si l'utilisateur Jorge souhaite utiliser la console pour modifier ses propres clés d'accès, il accède à la console IAM et choisit Users. Cette action indique à la console d'effectuer une demande `ListUsers`. Si Jorge n'a pas l'autorisation d'exécuter l'action `iam:ListUsers`, la console se voit refuser l'accès lorsqu'elle essaie de répertorier les utilisateurs.

Par conséquent, Jorge ne peut accéder à son propre nom et à ses propres clés d'accès, même s'il a l'autorisation d'exécuter les actions `CreateAccessKey` et `UpdateAccessKey`.

Si vous souhaitez autoriser les utilisateurs à administrer des groupes, des utilisateurs, des rôles, des politiques et des informations d'identification avec le AWS Management Console, vous devez inclure des autorisations pour les actions effectuées par la console. Pour obtenir des exemples de politiques que vous pouvez utiliser pour accorder ces autorisations à un utilisateur, consultez Exemples de politiques de gestion de ressources IAM.

Octroi d'autorisations dans plusieurs comptes AWS

Vous pouvez accorder directement à des utilisateurs IAM de votre propre compte un accès à vos ressources. Si des utilisateurs d'un autre compte ont besoin d'accéder à vos ressources, vous pouvez créer un rôle IAM qui est une entité incluant des autorisations, mais qui n'est pas associée à un utilisateur spécifique. Les utilisateurs d'autres comptes peuvent alors utiliser le rôle et accéder aux ressources selon les autorisations que vous avez attribuées à ce rôle. Pour plus d'informations, veuillez consulter Fournir l'accès à un utilisateur IAM dans un autre utilisateur Compte AWS dont vous êtes le propriétaire.

> **Note**
>
> Certains services prennent en charge les politiques basées sur les ressources, comme décrit dansPolitiques basées sur l'identité et Politiques basées sur une ressource (Amazon S3, Amazon SNS et Amazon SQS, par ex.). Pour ces services, une alternative à l'utilisation de rôles consiste à attacher une politique à la ressource (compartiment, rubrique ou file d'attente) que vous voulez partager. La politique basée sur les ressources peut spécifier le AWS compte autorisé à accéder à la ressource.

Autorisations pour qu'un service accède à un autre service

De nombreux AWS services accèdent à d'autres AWS services. Par exemple, plusieurs AWS services, dont Amazon EMR, Elastic Load Balancing et Amazon EC2 Auto Scaling, gèrent les instances Amazon EC2. D'autres AWS services utilisent les compartiments Amazon S3, les rubriques Amazon SNS, les files d'attente Amazon SQS, etc.

Le scénario pour gérer des autorisations dans ces cas varie selon le service. Voici des exemples de la façon dont les autorisations sont gérées pour différents services :

- Dans Amazon EC2 Auto Scaling, les utilisateurs doivent être autorisés à utiliser Auto Scaling, mais n'ont pas besoin de recevoir explicitement l'autorisation de gérer des instances Amazon EC2.
- Dans AWS Data Pipeline, un rôle IAM détermine ce que peut faire un pipeline ; les utilisateurs ont besoin d'une autorisation pour assumer ce rôle. (Pour obtenir des détails, veuillez consulter Granting Permissions to Pipelines with IAM (Octroi d'autorisations à des pipelines avec IAM)dans le Manuel du développeur AWS Data Pipeline .)

Pour plus de détails sur la manière de configurer correctement les autorisations afin qu'un AWS service puisse accomplir les tâches que vous souhaitez, reportez-vous à la documentation du service que vous appelez. Pour savoir comment créer un rôle pour un service, consultez Création d'un rôle pour la délégation d'autorisations à un service AWS.

Configuration d'un service avec un rôle IAM pour l'exécution d'une tâche en votre nom

Lorsque vous souhaitez configurer un AWS service pour qu'il fonctionne en votre nom, vous fournissez généralement l'ARN d'un rôle IAM qui définit ce que le service est autorisé à faire. AWS vérifie que vous êtes autorisé à transmettre un rôle à un service. Pour plus d’informations, consultez Octroi d'autorisations à un utilisateur pour transférer un rôle à un service AWS.

Actions requises

Les actions sont les choses que vous pouvez faire sur une ressource, comme l'affichage, la création, l'édition et la suppression de cette ressource. Les actions sont définies par chaque AWS service.

Pour autoriser une personne à effectuer une action, vous devez inclure les actions nécessaires dans une politique qui s'applique à l'identité appelante ou à la ressource affectée. En général, pour fournir l'autorisation requise pour effectuer une action, vous devez inclure cette action dans votre politique. Par exemple, pour créer un utilisateur, vous devez ajouter l' CreateUser action à votre politique.

Dans certains cas, une action peut exiger l'inclusion d'actions connexes supplémentaires dans votre politique. Par exemple, pour fournir à une personne l'autorisation de créer un annuaire dans AWS Directory Service à l'aide de l'opération `ds:CreateDirectory`, vous devez inclure les actions suivantes dans sa politique :

- `ds:CreateDirectory`
- `ec2:DescribeSubnets`
- `ec2:DescribeVpcs`

- `ec2:CreateSecurityGroup`
- `ec2:CreateNetworkInterface`
- `ec2:DescribeNetworkInterfaces`
- `ec2:AuthorizeSecurityGroupIngress`
- `ec2:AuthorizeSecurityGroupEgress`

Lorsque vous créez ou éditez une politique à l'aide de l'éditeur visuel, vous recevez des avertissements et des instructions pour vous aider à choisir toutes les actions requises pour votre politique.

Pour plus d'informations sur les autorisations requises pour créer un répertoire dans AWS Directory Service, voir Exemple 2 : Autoriser un utilisateur à créer un répertoire.

Exemples de politiques de gestion de ressources IAM

Voici des exemples de politiques IAM qui autorisent les utilisateurs à exécuter des tâches associées à la gestion des utilisateurs, des groupes et des informations d'identification IAM. Il s'agit notamment de politiques qui permettent aux utilisateurs de gérer leurs propres mots de passe, clés d'accès et dispositifs d'authentification multi-facteur (MFA).

Pour des exemples de politiques qui permettent aux utilisateurs d'effectuer des tâches avec d'autres AWS services, tels qu'Amazon S3, Amazon EC2 et DynamoDB, consultez. Exemples de politiques basées sur l'identité IAM

Rubriques

- Autoriser un utilisateur à répertorier les groupes, les utilisateurs et les politiques d'un compte, ainsi que d'autres informations à des fins d'élaboration de rapports
- Autoriser un utilisateur à gérer l'adhésion à un groupe
- Autoriser un utilisateur à gérer les utilisateurs IAM
- Autoriser les utilisateurs à définir la politique de mot de passe du compte
- Autoriser les utilisateurs à générer et extraire des rapports d'informations d'identification IAM
- Autoriser toutes les actions IAM (Accès Admin)

Autoriser un utilisateur à répertorier les groupes, les utilisateurs et les politiques d'un compte, ainsi que d'autres informations à des fins d'élaboration de rapports

La politique suivante permet à l'utilisateur d'appeler toute action IAM qui commence par la chaîne `Get` ou `List` et générer des rapports. Pour afficher l'exemple de politique, consultez IAM : autorise l'accès en lecture seule à la console IAM.

Autoriser un utilisateur à gérer l'adhésion à un groupe

La politique suivante permet à l'utilisateur de mettre à jour l'appartenance au groupe appelé MarketingGroup. Pour afficher l'exemple de politique, consultez IAM : autorise la gestion des membres d'un groupe par programmation et dans la console.

Autoriser un utilisateur à gérer les utilisateurs IAM

La politique suivante permet à un utilisateur d'exécuter toutes les tâches associées à la gestion des utilisateurs IAM, mais pas d'effectuer d'actions sur d'autres entités, par exemple la création de groupes ou de politiques. Les actions autorisées sont notamment :

- Création de l'utilisateur (action `CreateUser`).
- Suppression de l'utilisateur. Cette tâche requiert des autorisations pour effectuer toutes les actions suivantes : `DeleteSigningCertificate`, `DeleteLoginProfile`, `RemoveUserFromGroup` et `DeleteUser`.
- Affichage de la liste des utilisateurs du compte et des groupes (les actions `GetUser`, `ListUsers` et `ListGroupsForUser`).
- Affichage de la liste et suppression des politiques pour l'utilisateur (les actions `ListUserPolicies`, `ListAttachedUserPolicies`, `DetachUserPolicy` et `DeleteUserPolicy`)
- Changement de nom ou modification du chemin d'accès pour l'utilisateur (action `UpdateUser`). L'élément `Resource` doit inclure un ARN qui inclut à la fois le chemin d'accès source et le chemin d'accès cible. Pour plus d'informations sur les chemins d'accès, consultez Noms conviviaux et chemins.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowUsersToPerformUserActions",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:ListPolicies",
                "iam:GetPolicy",
                "iam:UpdateUser",
                "iam:AttachUserPolicy",
                "iam:ListEntitiesForPolicy",
                "iam:DeleteUserPolicy",
                "iam:DeleteUser",
                "iam:ListUserPolicies",
                "iam:CreateUser",
                "iam:RemoveUserFromGroup",
                "iam:AddUserToGroup",
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:PutUserPolicy",
                "iam:ListAttachedUserPolicies",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:DetachUserPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUsersToSeeStatsOnIAMConsoleDashboard",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccount*",
                "iam:ListAccount*"
            ],
            "Resource": "*"
        }
    ]
}
```

Plusieurs autorisations incluses dans la politique précédente permettent à l'utilisateur d'exécuter des tâches dans AWS Management Console. Les utilisateurs qui exécutent des tâches liées à l'utilisateur à partir de la AWS CLI, des kits SDK AWS ou de l'API Query HTTP IAM uniquement peuvent ne pas avoir besoin de certaines autorisations. Par exemple, si les utilisateurs connaissent déjà l'ARN des politiques à détacher d'un utilisateur, ils n'ont pas besoin d'autorisation `iam:ListAttachedUserPolicies`. La liste exacte des autorisations requises par un utilisateur varie en fonction des tâches qu'il doit effectuer lorsqu'il gère d'autres utilisateurs.

Les autorisations suivantes de la politique permettent l'accès aux tâches utilisateur via AWS Management Console:

- `iam:GetAccount*`
- `iam:ListAccount*`

Autoriser les utilisateurs à définir la politique de mot de passe du compte

Vous pouvez octroyer à certains utilisateurs des autorisations pour obtenir et mettre à jour la politique de mot de passe de votre Compte AWS. Pour afficher l'exemple de politique, consultez IAM : permet de définir les exigences de mot de passe du compte par programmation et dans la console.

Autoriser les utilisateurs à générer et extraire des rapports d'informations d'identification IAM

Vous pouvez autoriser les utilisateurs à générer et à télécharger un rapport répertoriant tous les utilisateurs de votre Compte AWS. Le rapport indique également l'état de plusieurs informations d'identification utilisateur, y compris les mots de passe, les clés d'accès, les dispositifs MFA et les certificats de signature. Pour de plus amples informations sur les rapports d'informations d'identification, veuillez consulter Obtenir des rapports d'informations d'identification pour votre Compte AWS. Pour afficher l'exemple de politique, consultez IAM : générer et extraire des rapports sur les informations d'identification IAM.

Autoriser toutes les actions IAM (Accès Admin)

Il est possible d'accorder à certains utilisateurs des autorisations administratives leur permettant d'effectuer toutes les actions dans IAM, y compris la gestion des mots de passe et des clés d'accès, des dispositifs MFA et des certificats utilisateur. L'exemple de politique suivant accorde ces autorisations.

> **Warning**
>
> Lorsque vous accordez à un utilisateur un accès complet à IAM, il n'y a aucune limite aux autorisations que l'utilisateur peut accorder à lui-même ou à d'autres. Ainsi, l'utilisateur peut créer de nouvelles entités IAM (utilisateurs ou rôles) et leur accorder un accès total à toutes les ressources de votre Compte AWS. Lorsque vous octroyez à un utilisateur un accès complet à IAM, vous lui donnez en réalité un accès total à toutes les ressources de votre Compte AWS. Cela inclut la possibilité de supprimer toutes les ressources. Vous

devez octroyer ces autorisations aux administrateurs approuvés uniquement, et vous devez également mettre en œuvre l'authentification multi-facteur (MFA) pour ces administrateurs.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*",
    "Resource": "*"
  }
}
```

Exemples de code pour IAM à l'aide AWS de kits de développement logiciel

Les exemples de code suivants montrent comment utiliser IAM avec un kit de développement AWS logiciel (SDK).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit de développement logiciel (SDK).

Exemples de code

- Exemples de code pour IAM utilisant des SDK AWS
  - Actions pour IAM à l'aide des SDK AWS
    - Utilisation AddClientIdToOpenIdConnectProvider avec un AWS SDK ou une CLI
    - Utilisation AddRoleToInstanceProfile avec un AWS SDK ou une CLI
    - Utilisation AddUserToGroup avec un AWS SDK ou une CLI
    - Utilisation AttachGroupPolicy avec un AWS SDK ou une CLI
    - Utilisation AttachRolePolicy avec un AWS SDK ou une CLI
    - Utilisation AttachUserPolicy avec un AWS SDK ou une CLI
    - Utilisation ChangePassword avec un AWS SDK ou une CLI
    - Utilisation CreateAccessKey avec un AWS SDK ou une CLI
    - Utilisation CreateAccountAlias avec un AWS SDK ou une CLI
    - Utilisation CreateGroup avec un AWS SDK ou une CLI
    - Utilisation CreateInstanceProfile avec un AWS SDK ou une CLI
    - Utilisation CreateLoginProfile avec un AWS SDK ou une CLI
    - Utilisation CreateOpenIdConnectProvider avec un AWS SDK ou une CLI
    - Utilisation CreatePolicy avec un AWS SDK ou une CLI
    - Utilisation CreatePolicyVersion avec un AWS SDK ou une CLI
    - Utilisation CreateRole avec un AWS SDK ou une CLI
    - Utilisation CreateSAMLProvider avec un AWS SDK ou une CLI
    - Utilisation CreateServiceLinkedRole avec un AWS SDK ou une CLI
    - Utilisation CreateUser avec un AWS SDK ou une CLI

- Utilisation CreateVirtualMfaDevice avec un AWS SDK ou une CLI
- Utilisation DeactivateMfaDevice avec un AWS SDK ou une CLI
- Utilisation DeleteAccessKey avec un AWS SDK ou une CLI
- Utilisation DeleteAccountAlias avec un AWS SDK ou une CLI
- Utilisation DeleteAccountPasswordPolicy avec un AWS SDK ou une CLI
- Utilisation DeleteGroup avec un AWS SDK ou une CLI
- Utilisation DeleteGroupPolicy avec un AWS SDK ou une CLI
- Utilisation DeleteInstanceProfile avec un AWS SDK ou une CLI
- Utilisation DeleteLoginProfile avec un AWS SDK ou une CLI
- Utilisation DeleteOpenIdConnectProvider avec un AWS SDK ou une CLI
- Utilisation DeletePolicy avec un AWS SDK ou une CLI
- Utilisation DeletePolicyVersion avec un AWS SDK ou une CLI
- Utilisation DeleteRole avec un AWS SDK ou une CLI
- Utilisation DeleteRolePermissionsBoundary avec un AWS SDK ou une CLI
- Utilisation DeleteRolePolicy avec un AWS SDK ou une CLI
- Utilisation DeleteSAMLProvider avec un AWS SDK ou une CLI
- Utilisation DeleteServerCertificate avec un AWS SDK ou une CLI
- Utilisation DeleteServiceLinkedRole avec un AWS SDK ou une CLI
- Utilisation DeleteSigningCertificate avec un AWS SDK ou une CLI
- Utilisation DeleteUser avec un AWS SDK ou une CLI
- Utilisation DeleteUserPermissionsBoundary avec un AWS SDK ou une CLI
- Utilisation DeleteUserPolicy avec un AWS SDK ou une CLI
- Utilisation DeleteVirtualMfaDevice avec un AWS SDK ou une CLI
- Utilisation DetachGroupPolicy avec un AWS SDK ou une CLI
- Utilisation DetachRolePolicy avec un AWS SDK ou une CLI
- Utilisation DetachUserPolicy avec un AWS SDK ou une CLI
- Utilisation EnableMfaDevice avec un AWS SDK ou une CLI
- Utilisation GenerateCredentialReport avec un AWS SDK ou une CLI
- Utilisation GenerateServiceLastAccessedDetails avec un AWS SDK ou une CLI
- Utilisation GetAccessKeyLastUsed avec un AWS SDK ou une CLI

- Utilisation GetAccountAuthorizationDetails avec un AWS SDK ou une CLI
- Utilisation GetAccountPasswordPolicy avec un AWS SDK ou une CLI
- Utilisation GetAccountSummary avec un AWS SDK ou une CLI
- Utilisation GetContextKeysForCustomPolicy avec un AWS SDK ou une CLI
- Utilisation GetContextKeysForPrincipalPolicy avec un AWS SDK ou une CLI
- Utilisation GetCredentialReport avec un AWS SDK ou une CLI
- Utilisation GetGroup avec un AWS SDK ou une CLI
- Utilisation GetGroupPolicy avec un AWS SDK ou une CLI
- Utilisation GetInstanceProfile avec un AWS SDK ou une CLI
- Utilisation GetLoginProfile avec un AWS SDK ou une CLI
- Utilisation GetOpenIdConnectProvider avec un AWS SDK ou une CLI
- Utilisation GetPolicy avec un AWS SDK ou une CLI
- Utilisation GetPolicyVersion avec un AWS SDK ou une CLI
- Utilisation GetRole avec un AWS SDK ou une CLI
- Utilisation GetRolePolicy avec un AWS SDK ou une CLI
- Utilisation GetSamlProvider avec un AWS SDK ou une CLI
- Utilisation GetServerCertificate avec un AWS SDK ou une CLI
- Utilisation GetServiceLastAccessedDetails avec un AWS SDK ou une CLI
- Utilisation GetServiceLastAccessedDetailsWithEntities avec un AWS SDK ou une CLI
- Utilisation GetServiceLinkedRoleDeletionStatus avec un AWS SDK ou une CLI
- Utilisation GetUser avec un AWS SDK ou une CLI
- Utilisation GetUserPolicy avec un AWS SDK ou une CLI
- Utilisation ListAccessKeys avec un AWS SDK ou une CLI
- Utilisation ListAccountAliases avec un AWS SDK ou une CLI
- Utilisation ListAttachedGroupPolicies avec un AWS SDK ou une CLI
- Utilisation ListAttachedRolePolicies avec un AWS SDK ou une CLI
- Utilisation ListAttachedUserPolicies avec un AWS SDK ou une CLI
- Utilisation ListEntitiesForPolicy avec un AWS SDK ou une CLI
- Utilisation ListGroupPolicies avec un AWS SDK ou une CLI

- Utilisation ListGroups avec un AWS SDK ou une CLI

- Utilisation ListGroupsForUser avec un AWS SDK ou une CLI
- Utilisation ListInstanceProfiles avec un AWS SDK ou une CLI
- Utilisation ListInstanceProfilesForRole avec un AWS SDK ou une CLI
- Utilisation ListMfaDevices avec un AWS SDK ou une CLI
- Utilisation ListOpenIdConnectProviders avec un AWS SDK ou une CLI
- Utilisation ListPolicies avec un AWS SDK ou une CLI
- Utilisation ListPolicyVersions avec un AWS SDK ou une CLI
- Utilisation ListRolePolicies avec un AWS SDK ou une CLI
- Utilisation ListRoleTags avec un AWS SDK ou une CLI
- Utilisation ListRoles avec un AWS SDK ou une CLI
- Utilisation ListSAMLProviders avec un AWS SDK ou une CLI
- Utilisation ListServerCertificates avec un AWS SDK ou une CLI
- Utilisation ListSigningCertificates avec un AWS SDK ou une CLI
- Utilisation ListUserPolicies avec un AWS SDK ou une CLI
- Utilisation ListUserTags avec un AWS SDK ou une CLI
- Utilisation ListUsers avec un AWS SDK ou une CLI
- Utilisation ListVirtualMfaDevices avec un AWS SDK ou une CLI
- Utilisation PutGroupPolicy avec un AWS SDK ou une CLI
- Utilisation PutRolePermissionsBoundary avec un AWS SDK ou une CLI
- Utilisation PutRolePolicy avec un AWS SDK ou une CLI
- Utilisation PutUserPermissionsBoundary avec un AWS SDK ou une CLI
- Utilisation PutUserPolicy avec un AWS SDK ou une CLI
- Utilisation RemoveClientIdFromOpenIdConnectProvider avec un AWS SDK ou une CLI
- Utilisation RemoveRoleFromInstanceProfile avec un AWS SDK ou une CLI
- Utilisation RemoveUserFromGroup avec un AWS SDK ou une CLI
- Utilisation ResyncMfaDevice avec un AWS SDK ou une CLI
- Utilisation SetDefaultPolicyVersion avec un AWS SDK ou une CLI
- Utilisation TagRole avec un AWS SDK ou une CLI
- Utilisation TagUser avec un AWS SDK ou une CLI
- Utilisation UntagRole avec un AWS SDK ou une CLI

- Utilisation UntagUser avec un AWS SDK ou une CLI
- Utilisation UpdateAccessKey avec un AWS SDK ou une CLI
- Utilisation UpdateAccountPasswordPolicy avec un AWS SDK ou une CLI
- Utilisation UpdateAssumeRolePolicy avec un AWS SDK ou une CLI
- Utilisation UpdateGroup avec un AWS SDK ou une CLI
- Utilisation UpdateLoginProfile avec un AWS SDK ou une CLI
- Utilisation UpdateOpenIdConnectProviderThumbprint avec un AWS SDK ou une CLI
- Utilisation UpdateRole avec un AWS SDK ou une CLI
- Utilisation UpdateRoleDescription avec un AWS SDK ou une CLI
- Utilisation UpdateSamlProvider avec un AWS SDK ou une CLI
- Utilisation UpdateServerCertificate avec un AWS SDK ou une CLI
- Utilisation UpdateSigningCertificate avec un AWS SDK ou une CLI
- Utilisation UpdateUser avec un AWS SDK ou une CLI
- Utilisation UploadServerCertificate avec un AWS SDK ou une CLI
- Utilisation UploadSigningCertificate avec un AWS SDK ou une CLI

- Scénarios pour IAM utilisant des SDK AWS
  - Créez et gérez un service résilient à l'aide d'un AWS SDK
  - Création d'un groupe IAM et ajout d'un utilisateur au groupe à l'aide d'un SDK AWS
  - Créez un utilisateur IAM et jouez un rôle dans AWS STS l'utilisation d'un SDK AWS
  - Créez des utilisateurs IAM en lecture seule et en lecture-écriture à l'aide d'un SDK AWS
  - Gérer les clés d'accès IAM à l'aide d'un SDK AWS
  - Gérer les politiques IAM à l'aide d'un SDK AWS
  - Gérer les rôles IAM à l'aide d'un SDK AWS
  - Gérez votre compte IAM à l'aide d'un SDK AWS
  - Restaurer une version d'une politique IAM à l'aide d'un SDK AWS
  - Utiliser l'API IAM Policy Builder à l'aide d'un SDK AWS

- Exemples de code pour AWS STS l'utilisation des AWS SDK
  - Actions relatives à AWS STS l'utilisation des AWS SDK
    - Utilisation AssumeRole avec un AWS SDK ou une CLI
    - Utilisation AssumeRoleWithWebIdentity avec un AWS SDK ou une CLI

- Utilisation DecodeAuthorizationMessage avec un AWS SDK ou une CLI
- Utilisation GetFederationToken avec un AWS SDK ou une CLI
- Utilisation GetSessionToken avec un AWS SDK ou une CLI

- Scénarios d' AWS STS utilisation des AWS SDK
  - Assumez un rôle IAM qui nécessite un jeton MFA à l' AWS STS aide d'un SDK AWS
  - Création d'une URL AWS STS pour les utilisateurs fédérés à l'aide d'un SDK AWS
  - Obtenez un jeton de session qui nécessite un jeton MFA à AWS STS l'aide d'un SDK AWS

Exemples de code pour IAM utilisant des SDK AWS

Les exemples de code suivants montrent comment utiliser IAM avec un kit de développement AWS logiciel (SDK).

Les actions sont des extraits de code de programmes plus larges et doivent être exécutées dans leur contexte. Alors que les actions vous indiquent comment appeler des fonctions de service individuelles, vous pouvez les voir en contexte dans leurs scénarios associés et dans des exemples interservices.

Les Scénarios sont des exemples de code qui vous montrent comment accomplir une tâche spécifique en appelant plusieurs fonctions au sein d'un même service.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Mise en route

Bonjour IAM

Les exemples de code suivants montrent comment démarrer avec IAM.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
namespace IAMActions;

public class HelloIAM
{
    static async Task Main(string[] args)
    {
        // Getting started with AWS Identity and Access Management (IAM). List
        // the policies for the account.
        var iamClient = new AmazonIdentityManagementServiceClient();

        var listPoliciesPaginator = iamClient.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        Console.WriteLine("Here are the policies defined for your account:\n");
        policies.ForEach(policy =>
        {
            Console.WriteLine($"Created:
 {policy.CreateDate}\t{policy.PolicyName}\t{policy.Description}");
        });
    }
}
```

- Pour plus de détails sur l'API, voir ListPolicieslа section Référence des AWS SDK for .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Code pour le MakeLists fichier CMake C.txt.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS iam)

# Set this project's name.
project("hello_iam")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
    # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.
```

```
    # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
 may need to uncomment this
    # and set the proper subdirectory to the executables' location.

    AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_iam.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Code pour le fichier source iam.cpp.

```
#include <aws/core/Aws.h>
#include <aws/iam/IAMClient.h>
#include <aws/iam/model/ListPoliciesRequest.h>
#include <iostream>
#include <iomanip>

/*
 *  A "Hello IAM" starter application which initializes an AWS Identity and
 Access Management (IAM) client
 *  and lists the IAM policies.
 *
 *  main function
 *
 *  Usage: 'hello_iam'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        const Aws::String DATE_FORMAT("%Y-%m-%d");
        Aws::Client::ClientConfiguration clientConfig;
```

```
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::IAM::IAMClient iamClient(clientConfig);
        Aws::IAM::Model::ListPoliciesRequest request;

        bool done = false;
        bool header = false;
        while (!done) {
            auto outcome = iamClient.ListPolicies(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Failed to list iam policies: " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = 1;
                break;
            }

            if (!header) {
                std::cout << std::left << std::setw(55) << "Name" <<
                          std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                          std::setw(64) << "Description" << std::setw(12) <<
                          "CreateDate" << std::endl;
                header = true;
            }

            const auto &policies = outcome.GetResult().GetPolicies();
            for (const auto &policy: policies) {
                std::cout << std::left << std::setw(55) <<
                          policy.GetPolicyName() << std::setw(30) <<
                          policy.GetPolicyId() << std::setw(80) <<
policy.GetArn() <<
                          std::setw(64) << policy.GetDescription() <<
std::setw(12) <<
                          policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
<<
                          std::endl;
            }

            if (outcome.GetResult().GetIsTruncated()) {
                request.SetMarker(outcome.GetResult().GetMarker());
            } else {
                done = true;
            }
        }
```

```
    }


    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- Pour plus de détails sur l'API, voir ListPolicieslа section Référence des AWS SDK for C++ API.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
package main

import (
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/iam"
)

// main uses the AWS SDK for Go (v2) to create an AWS Identity and Access
 Management (IAM)
// client and list up to 10 policies in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
```

```
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 iamClient := iam.NewFromConfig(sdkConfig)
 const maxPols = 10
 fmt.Printf("Let's list up to %v policies for your account.\n", maxPols)
 result, err := iamClient.ListPolicies(context.TODO(), &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPols),
 })
 if err != nil {
  fmt.Printf("Couldn't list policies for your account. Here's why: %v\n", err)
  return
 }
 if len(result.Policies) == 0 {
  fmt.Println("You don't have any policies!")
 } else {
  for _, policy := range result.Policies {
   fmt.Printf("\t%v\n", *policy.PolicyName)
  }
 }
}
```

- Pour plus de détails sur l'API, voir ListPolicieslа section Référence des AWS SDK for Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
```

```
import software.amazon.awssdk.services.iam.model.ListPoliciesResponse;
import software.amazon.awssdk.services.iam.model.Policy;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class HelloIAM {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listPolicies(iam);
    }

    public static void listPolicies(IamClient iam) {
        ListPoliciesResponse response = iam.listPolicies();
        List<Policy> polList = response.policies();
        polList.forEach(policy -> {
            System.out.println("Policy Name: " + policy.policyName());
        });
    }
}
```

- Pour plus de détails sur l'API, voir ListPoliciesla section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { IAMClient, paginateListPolicies } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listLocalPolicies = async () => {
  /**
   * In v3, the clients expose paginateOperationName APIs that are written using
 async generators so that you can use async iterators in a for await..of loop.
   * https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/index.html#paginators
   */
  const paginator = paginateListPolicies(
    { client, pageSize: 10 },
    // List only customer managed policies.
    { Scope: "Local" },
  );

  console.log("IAM policies defined in your account:");
  let policyCount = 0;
  for await (const page of paginator) {
    if (page.Policies) {
      page.Policies.forEach((p) => {
        console.log(`${p.PolicyName}`);
        policyCount++;
      });
    }
  }
  console.log(`Found ${policyCount} policies.`);
};
```

- Pour plus de détails sur l'API, voir ListPoliciesla section Référence des AWS SDK for JavaScript API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

De src/bin/hello.rs.

```
use aws_sdk_iam::error::SdkError;
use aws_sdk_iam::operation::list_policies::ListPoliciesError;
use clap::Parser;

const PATH_PREFIX_HELP: &str = "The path prefix for filtering the results.";

#[derive(Debug, clap::Parser)]
#[command(about)]
struct HelloScenarioArgs {
    #[arg(long, default_value="/", help=PATH_PREFIX_HELP)]
    pub path_prefix: String,
}

#[tokio::main]
async fn main() -> Result<(), SdkError<ListPoliciesError>> {
    let sdk_config = aws_config::load_from_env().await;
    let client = aws_sdk_iam::Client::new(&sdk_config);

    let args = HelloScenarioArgs::parse();

    iam_service::list_policies(client, args.path_prefix).await?;

    Ok(())
}
```

De src/iam-service-lib.rs.

```
pub async fn list_policies(
```

```
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
        .try_collect()
        .await?;

    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Pour plus de détails sur l'API, voir ListPolicieslа section de référence de l'API AWS SDK for Rust.

Exemples de code

- Actions pour IAM à l'aide des SDK AWS
  - Utilisation AddClientIdToOpenIdConnectProvider avec un AWS SDK ou une CLI
  - Utilisation AddRoleToInstanceProfile avec un AWS SDK ou une CLI
  - Utilisation AddUserToGroup avec un AWS SDK ou une CLI
  - Utilisation AttachGroupPolicy avec un AWS SDK ou une CLI
  - Utilisation AttachRolePolicy avec un AWS SDK ou une CLI
  - Utilisation AttachUserPolicy avec un AWS SDK ou une CLI

- Utilisation ChangePassword avec un AWS SDK ou une CLI
- Utilisation CreateAccessKey avec un AWS SDK ou une CLI
- Utilisation CreateAccountAlias avec un AWS SDK ou une CLI
- Utilisation CreateGroup avec un AWS SDK ou une CLI
- Utilisation CreateInstanceProfile avec un AWS SDK ou une CLI
- Utilisation CreateLoginProfile avec un AWS SDK ou une CLI
- Utilisation CreateOpenIdConnectProvider avec un AWS SDK ou une CLI
- Utilisation CreatePolicy avec un AWS SDK ou une CLI
- Utilisation CreatePolicyVersion avec un AWS SDK ou une CLI
- Utilisation CreateRole avec un AWS SDK ou une CLI
- Utilisation CreateSAMLProvider avec un AWS SDK ou une CLI
- Utilisation CreateServiceLinkedRole avec un AWS SDK ou une CLI
- Utilisation CreateUser avec un AWS SDK ou une CLI
- Utilisation CreateVirtualMfaDevice avec un AWS SDK ou une CLI
- Utilisation DeactivateMfaDevice avec un AWS SDK ou une CLI
- Utilisation DeleteAccessKey avec un AWS SDK ou une CLI
- Utilisation DeleteAccountAlias avec un AWS SDK ou une CLI
- Utilisation DeleteAccountPasswordPolicy avec un AWS SDK ou une CLI
- Utilisation DeleteGroup avec un AWS SDK ou une CLI
- Utilisation DeleteGroupPolicy avec un AWS SDK ou une CLI
- Utilisation DeleteInstanceProfile avec un AWS SDK ou une CLI
- Utilisation DeleteLoginProfile avec un AWS SDK ou une CLI
- Utilisation DeleteOpenIdConnectProvider avec un AWS SDK ou une CLI
- Utilisation DeletePolicy avec un AWS SDK ou une CLI
- Utilisation DeletePolicyVersion avec un AWS SDK ou une CLI
- Utilisation DeleteRole avec un AWS SDK ou une CLI
- Utilisation DeleteRolePermissionsBoundary avec un AWS SDK ou une CLI
- Utilisation DeleteRolePolicy avec un AWS SDK ou une CLI
- Utilisation DeleteSAMLProvider avec un AWS SDK ou une CLI
- Utilisation DeleteServerCertificate avec un AWS SDK ou une CLI

- Utilisation DeleteServiceLinkedRole avec un AWS SDK ou une CLI
- Utilisation DeleteSigningCertificate avec un AWS SDK ou une CLI
- Utilisation DeleteUser avec un AWS SDK ou une CLI
- Utilisation DeleteUserPermissionsBoundary avec un AWS SDK ou une CLI
- Utilisation DeleteUserPolicy avec un AWS SDK ou une CLI
- Utilisation DeleteVirtualMfaDevice avec un AWS SDK ou une CLI
- Utilisation DetachGroupPolicy avec un AWS SDK ou une CLI
- Utilisation DetachRolePolicy avec un AWS SDK ou une CLI
- Utilisation DetachUserPolicy avec un AWS SDK ou une CLI
- Utilisation EnableMfaDevice avec un AWS SDK ou une CLI
- Utilisation GenerateCredentialReport avec un AWS SDK ou une CLI
- Utilisation GenerateServiceLastAccessedDetails avec un AWS SDK ou une CLI
- Utilisation GetAccessKeyLastUsed avec un AWS SDK ou une CLI
- Utilisation GetAccountAuthorizationDetails avec un AWS SDK ou une CLI
- Utilisation GetAccountPasswordPolicy avec un AWS SDK ou une CLI
- Utilisation GetAccountSummary avec un AWS SDK ou une CLI
- Utilisation GetContextKeysForCustomPolicy avec un AWS SDK ou une CLI
- Utilisation GetContextKeysForPrincipalPolicy avec un AWS SDK ou une CLI
- Utilisation GetCredentialReport avec un AWS SDK ou une CLI
- Utilisation GetGroup avec un AWS SDK ou une CLI
- Utilisation GetGroupPolicy avec un AWS SDK ou une CLI
- Utilisation GetInstanceProfile avec un AWS SDK ou une CLI
- Utilisation GetLoginProfile avec un AWS SDK ou une CLI
- Utilisation GetOpenIdConnectProvider avec un AWS SDK ou une CLI
- Utilisation GetPolicy avec un AWS SDK ou une CLI
- Utilisation GetPolicyVersion avec un AWS SDK ou une CLI
- Utilisation GetRole avec un AWS SDK ou une CLI
- Utilisation GetRolePolicy avec un AWS SDK ou une CLI
- Utilisation GetSamlProvider avec un AWS SDK ou une CLI
- Utilisation GetServerCertificate avec un AWS SDK ou une CLI

- Utilisation GetServiceLastAccessedDetails avec un AWS SDK ou une CLI
- Utilisation GetServiceLastAccessedDetailsWithEntities avec un AWS SDK ou une CLI
- Utilisation GetServiceLinkedRoleDeletionStatus avec un AWS SDK ou une CLI
- Utilisation GetUser avec un AWS SDK ou une CLI
- Utilisation GetUserPolicy avec un AWS SDK ou une CLI
- Utilisation ListAccessKeys avec un AWS SDK ou une CLI
- Utilisation ListAccountAliases avec un AWS SDK ou une CLI
- Utilisation ListAttachedGroupPolicies avec un AWS SDK ou une CLI
- Utilisation ListAttachedRolePolicies avec un AWS SDK ou une CLI
- Utilisation ListAttachedUserPolicies avec un AWS SDK ou une CLI
- Utilisation ListEntitiesForPolicy avec un AWS SDK ou une CLI
- Utilisation ListGroupPolicies avec un AWS SDK ou une CLI
- Utilisation ListGroups avec un AWS SDK ou une CLI
- Utilisation ListGroupsForUser avec un AWS SDK ou une CLI
- Utilisation ListInstanceProfiles avec un AWS SDK ou une CLI
- Utilisation ListInstanceProfilesForRole avec un AWS SDK ou une CLI
- Utilisation ListMfaDevices avec un AWS SDK ou une CLI
- Utilisation ListOpenIdConnectProviders avec un AWS SDK ou une CLI
- Utilisation ListPolicies avec un AWS SDK ou une CLI
- Utilisation ListPolicyVersions avec un AWS SDK ou une CLI
- Utilisation ListRolePolicies avec un AWS SDK ou une CLI
- Utilisation ListRoleTags avec un AWS SDK ou une CLI
- Utilisation ListRoles avec un AWS SDK ou une CLI
- Utilisation ListSAMLProviders avec un AWS SDK ou une CLI
- Utilisation ListServerCertificates avec un AWS SDK ou une CLI
- Utilisation ListSigningCertificates avec un AWS SDK ou une CLI
- Utilisation ListUserPolicies avec un AWS SDK ou une CLI
- Utilisation ListUserTags avec un AWS SDK ou une CLI
- Utilisation ListUsers avec un AWS SDK ou une CLI
- Utilisation ListVirtualMfaDevices avec un AWS SDK ou une CLI

- Utilisation PutGroupPolicy avec un AWS SDK ou une CLI
- Utilisation PutRolePermissionsBoundary avec un AWS SDK ou une CLI
- Utilisation PutRolePolicy avec un AWS SDK ou une CLI
- Utilisation PutUserPermissionsBoundary avec un AWS SDK ou une CLI
- Utilisation PutUserPolicy avec un AWS SDK ou une CLI
- Utilisation RemoveClientIdFromOpenIdConnectProvider avec un AWS SDK ou une CLI
- Utilisation RemoveRoleFromInstanceProfile avec un AWS SDK ou une CLI
- Utilisation RemoveUserFromGroup avec un AWS SDK ou une CLI
- Utilisation ResyncMfaDevice avec un AWS SDK ou une CLI
- Utilisation SetDefaultPolicyVersion avec un AWS SDK ou une CLI
- Utilisation TagRole avec un AWS SDK ou une CLI
- Utilisation TagUser avec un AWS SDK ou une CLI
- Utilisation UntagRole avec un AWS SDK ou une CLI
- Utilisation UntagUser avec un AWS SDK ou une CLI
- Utilisation UpdateAccessKey avec un AWS SDK ou une CLI
- Utilisation UpdateAccountPasswordPolicy avec un AWS SDK ou une CLI
- Utilisation UpdateAssumeRolePolicy avec un AWS SDK ou une CLI
- Utilisation UpdateGroup avec un AWS SDK ou une CLI
- Utilisation UpdateLoginProfile avec un AWS SDK ou une CLI
- Utilisation UpdateOpenIdConnectProviderThumbprint avec un AWS SDK ou une CLI
- Utilisation UpdateRole avec un AWS SDK ou une CLI
- Utilisation UpdateRoleDescription avec un AWS SDK ou une CLI
- Utilisation UpdateSamlProvider avec un AWS SDK ou une CLI
- Utilisation UpdateServerCertificate avec un AWS SDK ou une CLI
- Utilisation UpdateSigningCertificate avec un AWS SDK ou une CLI
- Utilisation UpdateUser avec un AWS SDK ou une CLI
- Utilisation UploadServerCertificate avec un AWS SDK ou une CLI
- Utilisation UploadSigningCertificate avec un AWS SDK ou une CLI

- Scénarios pour IAM utilisant des SDK AWS
  - Créez et gérez un service résilient à l'aide d'un AWS SDK

- Création d'un groupe IAM et ajout d'un utilisateur au groupe à l'aide d'un SDK AWS
- Créez un utilisateur IAM et jouez un rôle dans AWS STS l'utilisation d'un SDK AWS
- Créez des utilisateurs IAM en lecture seule et en lecture-écriture à l'aide d'un SDK AWS
- Gérer les clés d'accès IAM à l'aide d'un SDK AWS
- Gérer les politiques IAM à l'aide d'un SDK AWS
- Gérer les rôles IAM à l'aide d'un SDK AWS
- Gérez votre compte IAM à l'aide d'un SDK AWS
- Restaurer une version d'une politique IAM à l'aide d'un SDK AWS
- Utiliser l'API IAM Policy Builder à l'aide d'un SDK AWS

Actions pour IAM à l'aide des SDK AWS

Les exemples de code suivants montrent comment effectuer des actions IAM individuelles avec des AWS SDK. Ces extraits appellent l'API IAM et sont des extraits de code de programmes plus volumineux qui doivent être exécutés en contexte. Chaque exemple inclut un lien vers GitHub, où vous pouvez trouver des instructions pour configurer et exécuter le code.

Les exemples suivants incluent uniquement les actions les plus couramment utilisées. Pour obtenir la liste complète, veuillez consulter la Référence d'API AWS Identity and Access Management (IAM).

Exemples

- Utilisation AddClientIdToOpenIdConnectProvider avec un AWS SDK ou une CLI
- Utilisation AddRoleToInstanceProfile avec un AWS SDK ou une CLI
- Utilisation AddUserToGroup avec un AWS SDK ou une CLI
- Utilisation AttachGroupPolicy avec un AWS SDK ou une CLI
- Utilisation AttachRolePolicy avec un AWS SDK ou une CLI
- Utilisation AttachUserPolicy avec un AWS SDK ou une CLI
- Utilisation ChangePassword avec un AWS SDK ou une CLI
- Utilisation CreateAccessKey avec un AWS SDK ou une CLI
- Utilisation CreateAccountAlias avec un AWS SDK ou une CLI
- Utilisation CreateGroup avec un AWS SDK ou une CLI
- Utilisation CreateInstanceProfile avec un AWS SDK ou une CLI

- Utilisation CreateLoginProfile avec un AWS SDK ou une CLI
- Utilisation CreateOpenIdConnectProvider avec un AWS SDK ou une CLI
- Utilisation CreatePolicy avec un AWS SDK ou une CLI
- Utilisation CreatePolicyVersion avec un AWS SDK ou une CLI
- Utilisation CreateRole avec un AWS SDK ou une CLI
- Utilisation CreateSAMLProvider avec un AWS SDK ou une CLI
- Utilisation CreateServiceLinkedRole avec un AWS SDK ou une CLI
- Utilisation CreateUser avec un AWS SDK ou une CLI
- Utilisation CreateVirtualMfaDevice avec un AWS SDK ou une CLI
- Utilisation DeactivateMfaDevice avec un AWS SDK ou une CLI
- Utilisation DeleteAccessKey avec un AWS SDK ou une CLI
- Utilisation DeleteAccountAlias avec un AWS SDK ou une CLI
- Utilisation DeleteAccountPasswordPolicy avec un AWS SDK ou une CLI
- Utilisation DeleteGroup avec un AWS SDK ou une CLI
- Utilisation DeleteGroupPolicy avec un AWS SDK ou une CLI
- Utilisation DeleteInstanceProfile avec un AWS SDK ou une CLI
- Utilisation DeleteLoginProfile avec un AWS SDK ou une CLI
- Utilisation DeleteOpenIdConnectProvider avec un AWS SDK ou une CLI
- Utilisation DeletePolicy avec un AWS SDK ou une CLI
- Utilisation DeletePolicyVersion avec un AWS SDK ou une CLI
- Utilisation DeleteRole avec un AWS SDK ou une CLI
- Utilisation DeleteRolePermissionsBoundary avec un AWS SDK ou une CLI
- Utilisation DeleteRolePolicy avec un AWS SDK ou une CLI
- Utilisation DeleteSAMLProvider avec un AWS SDK ou une CLI
- Utilisation DeleteServerCertificate avec un AWS SDK ou une CLI
- Utilisation DeleteServiceLinkedRole avec un AWS SDK ou une CLI
- Utilisation DeleteSigningCertificate avec un AWS SDK ou une CLI
- Utilisation DeleteUser avec un AWS SDK ou une CLI

- Utilisation DeleteUserPermissionsBoundary avec un AWS SDK ou une CLI
- Utilisation DeleteUserPolicy avec un AWS SDK ou une CLI
- Utilisation DeleteVirtualMfaDevice avec un AWS SDK ou une CLI
- Utilisation DetachGroupPolicy avec un AWS SDK ou une CLI
- Utilisation DetachRolePolicy avec un AWS SDK ou une CLI
- Utilisation DetachUserPolicy avec un AWS SDK ou une CLI
- Utilisation EnableMfaDevice avec un AWS SDK ou une CLI
- Utilisation GenerateCredentialReport avec un AWS SDK ou une CLI
- Utilisation GenerateServiceLastAccessedDetails avec un AWS SDK ou une CLI
- Utilisation GetAccessKeyLastUsed avec un AWS SDK ou une CLI
- Utilisation GetAccountAuthorizationDetails avec un AWS SDK ou une CLI
- Utilisation GetAccountPasswordPolicy avec un AWS SDK ou une CLI
- Utilisation GetAccountSummary avec un AWS SDK ou une CLI
- Utilisation GetContextKeysForCustomPolicy avec un AWS SDK ou une CLI
- Utilisation GetContextKeysForPrincipalPolicy avec un AWS SDK ou une CLI
- Utilisation GetCredentialReport avec un AWS SDK ou une CLI
- Utilisation GetGroup avec un AWS SDK ou une CLI
- Utilisation GetGroupPolicy avec un AWS SDK ou une CLI
- Utilisation GetInstanceProfile avec un AWS SDK ou une CLI
- Utilisation GetLoginProfile avec un AWS SDK ou une CLI
- Utilisation GetOpenIdConnectProvider avec un AWS SDK ou une CLI
- Utilisation GetPolicy avec un AWS SDK ou une CLI
- Utilisation GetPolicyVersion avec un AWS SDK ou une CLI
- Utilisation GetRole avec un AWS SDK ou une CLI
- Utilisation GetRolePolicy avec un AWS SDK ou une CLI
- Utilisation GetSamlProvider avec un AWS SDK ou une CLI
- Utilisation GetServerCertificate avec un AWS SDK ou une CLI
- Utilisation GetServiceLastAccessedDetails avec un AWS SDK ou une CLI

- Utilisation GetServiceLastAccessedDetailsWithEntities avec un AWS SDK ou une CLI
- Utilisation GetServiceLinkedRoleDeletionStatus avec un AWS SDK ou une CLI
- Utilisation GetUser avec un AWS SDK ou une CLI
- Utilisation GetUserPolicy avec un AWS SDK ou une CLI
- Utilisation ListAccessKeys avec un AWS SDK ou une CLI
- Utilisation ListAccountAliases avec un AWS SDK ou une CLI
- Utilisation ListAttachedGroupPolicies avec un AWS SDK ou une CLI
- Utilisation ListAttachedRolePolicies avec un AWS SDK ou une CLI
- Utilisation ListAttachedUserPolicies avec un AWS SDK ou une CLI
- Utilisation ListEntitiesForPolicy avec un AWS SDK ou une CLI
- Utilisation ListGroupPolicies avec un AWS SDK ou une CLI
- Utilisation ListGroups avec un AWS SDK ou une CLI
- Utilisation ListGroupsForUser avec un AWS SDK ou une CLI
- Utilisation ListInstanceProfiles avec un AWS SDK ou une CLI
- Utilisation ListInstanceProfilesForRole avec un AWS SDK ou une CLI
- Utilisation ListMfaDevices avec un AWS SDK ou une CLI
- Utilisation ListOpenIdConnectProviders avec un AWS SDK ou une CLI
- Utilisation ListPolicies avec un AWS SDK ou une CLI
- Utilisation ListPolicyVersions avec un AWS SDK ou une CLI
- Utilisation ListRolePolicies avec un AWS SDK ou une CLI
- Utilisation ListRoleTags avec un AWS SDK ou une CLI
- Utilisation ListRoles avec un AWS SDK ou une CLI
- Utilisation ListSAMLProviders avec un AWS SDK ou une CLI
- Utilisation ListServerCertificates avec un AWS SDK ou une CLI
- Utilisation ListSigningCertificates avec un AWS SDK ou une CLI
- Utilisation ListUserPolicies avec un AWS SDK ou une CLI
- Utilisation ListUserTags avec un AWS SDK ou une CLI
- Utilisation ListUsers avec un AWS SDK ou une CLI

- Utilisation ListVirtualMfaDevices avec un AWS SDK ou une CLI
- Utilisation PutGroupPolicy avec un AWS SDK ou une CLI
- Utilisation PutRolePermissionsBoundary avec un AWS SDK ou une CLI
- Utilisation PutRolePolicy avec un AWS SDK ou une CLI
- Utilisation PutUserPermissionsBoundary avec un AWS SDK ou une CLI
- Utilisation PutUserPolicy avec un AWS SDK ou une CLI
- Utilisation RemoveClientIdFromOpenIdConnectProvider avec un AWS SDK ou une CLI
- Utilisation RemoveRoleFromInstanceProfile avec un AWS SDK ou une CLI
- Utilisation RemoveUserFromGroup avec un AWS SDK ou une CLI
- Utilisation ResyncMfaDevice avec un AWS SDK ou une CLI
- Utilisation SetDefaultPolicyVersion avec un AWS SDK ou une CLI
- Utilisation TagRole avec un AWS SDK ou une CLI
- Utilisation TagUser avec un AWS SDK ou une CLI
- Utilisation UntagRole avec un AWS SDK ou une CLI
- Utilisation UntagUser avec un AWS SDK ou une CLI
- Utilisation UpdateAccessKey avec un AWS SDK ou une CLI
- Utilisation UpdateAccountPasswordPolicy avec un AWS SDK ou une CLI
- Utilisation UpdateAssumeRolePolicy avec un AWS SDK ou une CLI
- Utilisation UpdateGroup avec un AWS SDK ou une CLI
- Utilisation UpdateLoginProfile avec un AWS SDK ou une CLI
- Utilisation UpdateOpenIdConnectProviderThumbprint avec un AWS SDK ou une CLI
- Utilisation UpdateRole avec un AWS SDK ou une CLI
- Utilisation UpdateRoleDescription avec un AWS SDK ou une CLI
- Utilisation UpdateSamlProvider avec un AWS SDK ou une CLI
- Utilisation UpdateServerCertificate avec un AWS SDK ou une CLI
- Utilisation UpdateSigningCertificate avec un AWS SDK ou une CLI
- Utilisation UpdateUser avec un AWS SDK ou une CLI
- Utilisation UploadServerCertificate avec un AWS SDK ou une CLI
- Utilisation UploadSigningCertificate avec un AWS SDK ou une CLI

Utilisation **`AddClientIdToOpenIdConnectProvider`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`AddClientIdToOpenIdConnectProvider`.

CLI

AWS CLI

Pour ajouter un identifiant client (audience) à un fournisseur Open-ID Connect (OIDC)

La `add-client-id-to-open-id-connect-provider` commande suivante ajoute l'ID client `my-application-ID` au fournisseur OIDC nommé`server.example.com`.

```
aws iam add-client-id-to-open-id-connect-provider \
    --client-id my-application-ID \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

Cette commande ne produit aucun résultat.

Pour créer un fournisseur OIDC, utilisez la `create-open-id-connect-provider` commande.

Pour plus d'informations, consultez la section Création de fournisseurs d'identité OpenID Connect (OIDC) dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous AddClientIdToOpenIdConnectProviderà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cette commande ajoute l'ID client (ou audience) **`my-application-ID`** au fournisseur OIDC existant nommé**`server.example.com`**.

```
Add-IAMClientIDToOpenIDConnectProvider -ClientID "my-application-ID"
 -OpenIDConnectProviderARN "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
```

- Pour plus de détails sur l'API, reportez-vous AddClientIdToOpenIdConnectProviderà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`AddRoleToInstanceProfile`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`AddRoleToInstanceProfile`.

CLI

AWS CLI

Pour ajouter un rôle à un profil d'instance

La `add-role-to-instance-profile` commande suivante ajoute le rôle nommé `S3Access` au profil d'instance nommé`Webserver`.

```
aws iam add-role-to-instance-profile \
    --role-name S3Access \
    --instance-profile-name Webserver
```

Cette commande ne produit aucun résultat.

Pour créer un profil d'instance, utilisez la `create-instance-profile` commande.

Pour plus d’informations, consultez Utilisation d’un rôle IAM pour accorder des autorisations à des applications s’exécutant sur des instances Amazon EC2 dans le AWS Guide de l’utilisateur IAM.

- Pour plus de détails sur l'API, consultez la section AddRoleToInstanceProfil dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cette commande ajoute le rôle nommé **`S3Access`** à un profil d'instance existant nommé**`webserver`**. Pour créer le profil d'instance, utilisez la **`New-IAMInstanceProfile`**

commande. Après avoir créé le profil d'instance et l'avoir associé à un rôle à l'aide de cette commande, vous pouvez l'associer à une instance EC2. Pour ce faire, utilisez l'**New-EC2Instance**applet de commande avec le paramètre **InstanceProfile_Arn** ou le **InstanceProfile-Name** paramètre pour lancer la nouvelle instance.

```
Add-IAMRoleToInstanceProfile -RoleName "S3Access" -InstanceProfileName
 "webserver"
```

- Pour plus de détails sur l'API, consultez la section AddRoleToInstanceProfil dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **AddUserToGroup** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`AddUserToGroup`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Créer un groupe et ajouter un utilisateur

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
```

```
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous AddUserToGroupà la section Référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour ajouter un utilisateur à un groupe IAM

La commande `add-user-to-group` suivante ajoute un utilisateur IAM nommé `Bob` au groupe IAM nommé `Admins`.

```
aws iam add-user-to-group \
    --user-name Bob \
    --group-name Admins
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, veuillez consulter Ajout et suppression d'utilisateurs dans un groupe IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous AddUserToGroupà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cette commande ajoute l'utilisateur nommé **Bob** au groupe nomméAdmins.

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- Pour plus de détails sur l'API, consultez la section AddUserToGroupRéférence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation `AttachGroupPolicy` avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`AttachGroupPolicy`.

CLI

AWS CLI

Pour associer une politique gérée à un groupe IAM

La `attach-group-policy` commande suivante associe la politique AWS gérée nommée `ReadOnlyAccess` au groupe IAM nommé`Finance`.

```
aws iam attach-group-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --group-name Finance
```

Cette commande ne produit aucun résultat.

Pour plus d’informations, consultez Politiques gérées et politiques en ligne dans le Guide de l’utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez AttachGroupla section Politique dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple associe la politique gérée par le client nommée **`TesterPolicy`** au groupe **`Testers`** IAM. Les utilisateurs de ce groupe sont immédiatement affectés par les autorisations définies dans la version par défaut de cette politique.

```
Register-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Exemple 2 : Cet exemple associe la politique AWS gérée nommée **`AdministratorAccess`** au groupe **`Admins`** IAM. Les utilisateurs de ce groupe sont immédiatement affectés par les autorisations définies dans la dernière version de cette politique.

```
Register-IAMGroupPolicy -GroupName Admins -PolicyArn arn:aws:iam::aws:policy/
AdministratorAccess
```

- Pour plus de détails sur l'API, consultez AttachGroupla section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`AttachRolePolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`AttachRolePolicy`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Créer un groupe et ajouter un utilisateur
- Créer un utilisateur et assumer d'un rôle
- Gérer les rôles

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, consultez AttachRolela section Politique dans le Guide de référence des AWS SDK for .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
```

```
  echo "  -p policy_ARN -- The IAM policy document ARN."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy ARN with the -p parameter."
  usage
  return 1
fi

response=$(aws iam attach-role-policy \
  --role-name "$role_name" \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, consultez AttachRolela section Politique dans AWS CLI Command Reference.

C++

SDK pour C++

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::attachRolePolicy(const Aws::String &roleName,
                                   const Aws::String &policyArn,
                                   const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::ListAttachedRolePoliciesRequest list_request;
    list_request.SetRoleName(roleName);

    bool done = false;
    while (!done) {
        auto list_outcome = iam.ListAttachedRolePolicies(list_request);
        if (!list_outcome.IsSuccess()) {
            std::cerr << "Failed to list attached policies of role " <<
                      roleName << ": " << list_outcome.GetError().GetMessage() <<
                      std::endl;
            return false;
        }

        const auto &policies = list_outcome.GetResult().GetAttachedPolicies();
```

```
        if (std::any_of(policies.cbegin(), policies.cend(),
                        [=](const Aws::IAM::Model::AttachedPolicy &policy) {
                                return policy.GetPolicyArn() == policyArn;
                        })) {
            std::cout << "Policy " << policyArn <<
                      " is already attached to role " << roleName << std::endl;
            return true;
        }

        done = !list_outcome.GetResult().GetIsTruncated();
        list_request.SetMarker(list_outcome.GetResult().GetMarker());
    }

    Aws::IAM::Model::AttachRolePolicyRequest request;
    request.SetRoleName(roleName);
    request.SetPolicyArn(policyArn);

    Aws::IAM::Model::AttachRolePolicyOutcome outcome =
 iam.AttachRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to attach policy " << policyArn << " to role " <<
                  roleName << ": " << outcome.GetError().GetMessage() <<
 std::endl;
    }
    else {
        std::cout << "Successfully attached policy " << policyArn << " to role "
 <<
                  roleName << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, consultez AttachRolela section Politique dans le Guide de référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour attacher une politique gérée à un rôle IAM

La `attach-role-policy` commande suivante associe la politique AWS gérée nommée `ReadOnlyAccess` au rôle IAM nomméReadOnlyRole.

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --role-name ReadOnlyRole
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez Politiques gérées et politiques en ligne dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez AttachRolela section Politique dans AWS CLI Command Reference.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(policyArn string, roleName string)
 error {
 _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),
 &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
```

```
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}
```

- Pour plus de détails sur l'API, consultez AttachRolela section Politique dans le Guide de référence des AWS SDK for Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.AttachRolePolicyRequest;
import software.amazon.awssdk.services.iam.model.AttachedPolicy;
import software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesRequest;
import
 software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesResponse;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
```

```
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AttachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleName = args[0];
        String policyArn = args[1];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        attachIAMRolePolicy(iam, roleName, policyArn);
        iam.close();
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
 policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
 ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();

            ListAttachedRolePoliciesResponse response =
 iam.listAttachedRolePolicies(request);
```

```
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();

            // Ensure that the policy is not attached to this role
            String polArn = "";
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
 this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
 AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);

            System.out.println("Successfully attached policy " + policyArn +
                    " to role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Pour plus de détails sur l'API, consultez AttachRolela section Politique dans le Guide de référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Attachez la politique.

```
import { AttachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const attachRolePolicy = (policyArn, roleName) => {
  const command = new AttachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez AttachRolela section Politique dans le Guide de référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        console.log(
          "AmazonDynamoDBFullAccess is already attached to this role."
        );
        process.exit();
      }
    });
    var params = {
      PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
      RoleName: process.argv[2],
    };
    iam.attachRolePolicy(params, function (err, data) {
      if (err) {
        console.log("Unable to attach policy to role", err);
      } else {
        console.log("Role attached successfully");
```

```
      }
    });
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez AttachRolela section Politique dans le Guide de référence des AWS SDK for JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun attachIAMRolePolicy(
    roleNameVal: String,
    policyArnVal: String
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
```

```
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}
```

- Pour plus de détails sur l'API, consultez AttachRolela section Politique du AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

    public function attachRolePolicy($roleName, $policyArn)
    {
        return $this->customWaiter(function () use ($roleName, $policyArn) {
            $this->iamClient->attachRolePolicy([
                'PolicyArn' => $policyArn,
                'RoleName' => $roleName,
            ]);
        });
    }
```

- Pour plus de détails sur l'API, consultez AttachRolela section Politique dans le Guide de référence des AWS SDK for PHP API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple associe la politique AWS gérée nommée **`SecurityAudit`** au rôle **`CoSecurityAuditors`** IAM. Les utilisateurs qui assument ce rôle sont immédiatement affectés par les autorisations définies dans la dernière version de cette politique.

```
Register-IAMRolePolicy -RoleName CoSecurityAuditors -PolicyArn
 arn:aws:iam::aws:policy/SecurityAudit
```

- Pour plus de détails sur l'API, consultez AttachRolela section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Attacher une politique à un rôle à l'aide de l'objet de politique Boto3.

```
def attach_to_role(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).attach_role(RoleName=role_name)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise
```

Attacher une politique à un rôle à l'aide de l'objet de rôle Boto3.

```
def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise
```

- Pour plus de détails sur l'API, consultez le AttachRoledocument de référence de l'API Policy in AWS SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, attache et détache les politiques de rôle.

```
# Manages policies in AWS Identity and Access Management (IAM)
```

```
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
 #{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
 exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
 #{e.message}")
    raise
  end
```

```
# Attaches a policy to a role
#
# @param role_name [String] The name of the role
# @param policy_arn [String] The policy ARN
# @return [Boolean] true if successful, false otherwise
def attach_policy_to_role(role_name, policy_arn)
  @iam_client.attach_role_policy(
    role_name: role_name,
    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error attaching policy to role: #{e.message}")
  false
end

# Lists policy ARNs attached to a role
#
# @param role_name [String] The name of the role
# @return [Array<String>] List of policy ARNs
def list_attached_policy_arns(role_name)
  response = @iam_client.list_attached_role_policies(role_name: role_name)
  response.attached_policies.map(&:policy_arn)
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing policies attached to role: #{e.message}")
  []
end

# Detaches a policy from a role
#
# @param role_name [String] The name of the role
# @param policy_arn [String] The policy ARN
# @return [Boolean] true if successful, false otherwise
def detach_policy_from_role(role_name, policy_arn)
  @iam_client.detach_role_policy(
    role_name: role_name,
    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error detaching policy from role: #{e.message}")
  false
end
```

```
end
```

- Pour plus de détails sur l'API, consultez AttachRolela section Politique dans le Guide de référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn attach_role_policy(
    client: &iamClient,
    role: &Role,
    policy: &Policy,
) -> Result<AttachRolePolicyOutput, SdkError<AttachRolePolicyError>> {
    client
        .attach_role_policy()
        .role_name(role.role_name())
        .policy_arn(policy.arn().unwrap_or_default())
        .send()
        .await
}
```

- Pour plus de détails sur l'API, consultez AttachRolela section Politique du AWS SDK pour la référence de l'API Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
public func attachRolePolicy(role: String, policyArn: String) async throws {
    let input = AttachRolePolicyInput(
        policyArn: policyArn,
        roleName: role
    )
    do {
        _ = try await client.attachRolePolicy(input: input)
    } catch {
        throw error
    }
}
```

- Pour plus de détails sur l'API, consultez AttachRolela section Politique du AWS SDK pour la référence à l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`AttachUserPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`AttachUserPolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Créer des utilisateurs en lecture seule et en lecture-écriture

CLI

AWS CLI

Pour attacher une politique gérée à un utilisateur IAM

La `attach-user-policy` commande suivante associe la politique AWS gérée nommée `AdministratorAccess` à l'utilisateur IAM nomméAlice.

```
aws iam attach-user-policy \
    --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
    --user-name Alice
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez Politiques gérées et politiques en ligne dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez AttachUserla section Politique dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple attache la politique AWS gérée nommée **`AmazonCognitoPowerUser`** à l'utilisateur **`Bob`** IAM. L'utilisateur est immédiatement affecté par les autorisations définies dans la dernière version de cette politique.

```
Register-IAMUserPolicy -UserName Bob -PolicyArn arn:aws:iam::aws:policy/
AmazonCognitoPowerUser
```

- Pour plus de détails sur l'API, consultez AttachUserla section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
 user_name)
        raise
```

- Pour plus de détails sur l'API, consultez le AttachUserdocument de référence de l'API Policy in AWS SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
  # Attaches a policy to a user
  #
  # @param user_name [String] The name of the user
  # @param policy_arn [String] The Amazon Resource Name (ARN) of the policy
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_user(user_name, policy_arn)
    @iam_client.attach_user_policy(
      user_name: user_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to user: #{e.message}")
    false
  end
```

- Pour plus de détails sur l'API, consultez AttachUserla section Politique dans le Guide de référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn attach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .attach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;
```

```
    Ok(())
}
```

- Pour plus de détails sur l'API, consultez AttachUserla section Politique du AWS SDK pour la référence de l'API Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ChangePassword`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ChangePassword`.

CLI

AWS CLI

Pour modifier le mot de passe de votre utilisateur IAM

Pour modifier le mot de passe de votre utilisateur IAM, nous vous recommandons d'utiliser le `--cli-input-json` paramètre pour transmettre un fichier JSON contenant vos anciens et nouveaux mots de passe. Grâce à cette méthode, vous pouvez utiliser des mots de passe forts contenant des caractères non alphanumériques. Il peut être difficile d'utiliser des mots de passe contenant des caractères non alphanumériques lorsque vous les transmettez en tant que paramètres de ligne de commande. Pour utiliser le `--cli-input-json` paramètre, commencez par utiliser la `change-password` commande avec le `--generate-cli-skeleton` paramètre, comme dans l'exemple suivant.

```
aws iam change-password \
    --generate-cli-skeleton > change-password.json
```

La commande précédente crée un fichier JSON appelé change-password.json que vous pouvez utiliser pour renseigner vos anciens et nouveaux mots de passe. Par exemple, le fichier peut ressembler à ce qui suit.

```
{
```

```
    "OldPassword": "3s0K_;xh4~8XXI",
    "NewPassword": "]35d/{pB9Fo9wJ"
}
```

Ensuite, pour modifier votre mot de passe, réutilisez la `change-password` commande, en passant cette fois le `--cli-input-json` paramètre pour spécifier votre fichier JSON. La `change-password` commande suivante utilise le `--cli-input-json` paramètre avec un fichier JSON appelé change-password.json.

```
aws iam change-password \
    --cli-input-json file://change-password.json
```

Cette commande ne produit aucun résultat.

Cette commande ne peut être appelée que par les utilisateurs IAM. Si cette commande est appelée à l'aide des informations d'identification du AWS compte (root), elle renvoie une `InvalidUserType` erreur.

Pour plus d'informations, consultez la section Comment un utilisateur IAM modifie son propre mot de passe dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ChangePasswordà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cette commande modifie le mot de passe de l'utilisateur qui exécute la commande. Cette commande ne peut être appelée que par les utilisateurs IAM. Si cette commande est appelée lorsque vous êtes connecté avec les informations d'identification du AWS compte (root), elle renvoie une erreur. **`InvalidUserType`**

```
Edit-IAMPassword -OldPassword "MyOldP@ssw0rd" -NewPassword "MyNewP@ssw0rd"
```

- Pour plus de détails sur l'API, reportez-vous ChangePasswordà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreateAccessKey`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreateAccessKey`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Créer un groupe et ajouter un utilisateur
- Créer un utilisateur et assumer d'un rôle
- Créer des utilisateurs en lecture seule et en lecture-écriture
- Gérer des clés d'accès

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
CreateAccessKeyRequest
        {
            UserName = userName,
        });
```

```
        return response.AccessKey;

    }
```

- Pour plus de détails sur l'API, voir CreateAccessKey in AWS SDK for .NET API Reference.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
```

```
##############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}
```

- Pour plus de détails sur l'API, voir CreateAccessKey in AWS CLI Command Reference.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
Aws::String AwsDoc::IAM::createAccessKey(const Aws::String &userName,
                                         const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateAccessKeyRequest request;
    request.SetUserName(userName);
```

```
    Aws::String result;
    Aws::IAM::Model::CreateAccessKeyOutcome outcome =
 iam.CreateAccessKey(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating access key for IAM user " << userName
                  << ":" << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &accessKey = outcome.GetResult().GetAccessKey();
        std::cout << "Successfully created access key for IAM user " <<
                  userName << std::endl << "  aws_access_key_id = " <<
                  accessKey.GetAccessKeyId() << std::endl <<
                  " aws_secret_access_key = " << accessKey.GetSecretAccessKey()
 <<
                  std::endl;
        result = accessKey.GetAccessKeyId();
    }

    return result;
}
```

- Pour plus de détails sur l'API, voir CreateAccessKey in AWS SDK for C++ API Reference.

CLI

AWS CLI

Pour créer une clé d’accès pour un utilisateur IAM

La commande `create-access-key` suivante créer une clé d’accès (un ID de clé d’accès et une clé d’accès secrète) pour l’utilisateur IAM nommé Bob.

```
aws iam create-access-key \
    --user-name Bob
```

Sortie :

```
{
    "AccessKey": {
        "UserName": "Bob",
        "Status": "Active",
```

```
        "CreateDate": "2015-03-09T18:39:23.411Z",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

Conservez votre clé d'accès secrète dans un emplacement sécurisé. Si elle est perdue, elle ne peut pas être récupérée et vous devez créer une nouvelle clé.

Pour de plus amples informations, veuillez consulter Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir CreateAccessKey in AWS CLI Command Reference.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(userName string)
 (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
 &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
```

```
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}
```

- Pour plus de détails sur l'API, voir CreateAccessKey in AWS SDK for Go API Reference.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.CreateAccessKeyRequest;
import software.amazon.awssdk.services.iam.model.CreateAccessKeyResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAccessKey {
    public static void main(String[] args) {
        final String usage = """
```

```
                Usage:
                   <user>\s

                Where:
                   user - An AWS IAM user that you can obtain from the AWS
 Management Console.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String user = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String keyId = createIAMAccessKey(iam, user);
        System.out.println("The Key Id is " + keyId);
        iam.close();
    }

    public static String createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey().accessKeyId();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Pour plus de détails sur l'API, voir CreateAccessKey in AWS SDK for Java 2.x API Reference.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez la clé d'accès.

```
import { CreateAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 */
export const createAccessKey = (userName) => {
  const command = new CreateAccessKeyCommand({ UserName: userName });
  return client.send(command);
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, voir CreateAccessKey in AWS SDK for JavaScript API Reference.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccessKey({ UserName: "IAM_USER_NAME" }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.AccessKey);
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, voir CreateAccessKey in AWS SDK for JavaScript API Reference.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun createIAMAccessKey(user: String?): String {
    val request =
        CreateAccessKeyRequest {
            userName = user
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createAccessKey(request)
```

```
        return response.accessKey?.accessKeyId.toString()
    }
}
```

- Pour plus de détails sur l'API, voir CreateAccessKey in AWS SDK for Kotlin API reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée une nouvelle paire de clés d'accès et de clés d'accès secrètes et les attribue à l'utilisateur**David**. Assurez-vous d'enregistrer les **SecretAccessKey** valeurs **AccessKeyId** et dans un fichier, car c'est la seule fois où vous pouvez obtenir les**SecretAccessKey**. Vous ne pourrez pas la récupérer ultérieurement. Si vous perdez la clé secrète, vous devez créer une nouvelle paire de clés d'accès.

```
New-IAMAccessKey -UserName David
```

Sortie :

```
AccessKeyId     : AKIAIOSFODNN7EXAMPLE
CreateDate      : 4/13/2015 1:00:42 PM
SecretAccessKey : wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Status          : Active
UserName        : David
```

- Pour plus de détails sur l'API, consultez CreateAccessla section Key in AWS Tools for PowerShell Cmdlet Reference.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair
```

- Pour plus de détails sur l'API, consultez le manuel de référence de l'API CreateAccessKey in AWS SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, désactive et supprime les clés d'accès.

```
# Manages access keys for IAM users
class AccessKeyManager
```

```
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "AccessKeyManager"
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity => e
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded => e
    @logger.error("Error creating access key: limit exceeded. Cannot create
more.")
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
```

```
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: "Inactive"
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, voir CreateAccessKey in AWS SDK for Ruby API Reference.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn create_access_key(client: &iamClient, user_name: &str) ->
 Result<AccessKey, iamError> {
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;

    let response: Result<CreateAccessKeyOutput, SdkError<CreateAccessKeyError>> =
 loop {
        match client.create_access_key().user_name(user_name).send().await {
            Ok(inner_response) => {
                break Ok(inner_response);
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    Ok(response.unwrap().access_key.unwrap())
}
```

- Pour plus de détails sur l'API, voir CreateAccessKey in AWS SDK for Rust API reference.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   public func createAccessKey(userName: String) async throws ->
IAMClientTypes.AccessKey {
       let input = CreateAccessKeyInput(
           userName: userName
       )
       do {
           let output = try await iamClient.createAccessKey(input: input)
           guard let accessKey = output.accessKey else {
               throw ServiceHandlerError.keyError
           }
           return accessKey
       } catch {
           throw error
       }
   }
```

- Pour plus de détails sur l'API, voir CreateAccessKey in AWS SDK for Swift API reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreateAccountAlias`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreateAccountAlias`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer votre compte

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::createAccountAlias(const Aws::String &aliasName,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::CreateAccountAliasRequest request;
    request.SetAccountAlias(aliasName);

    Aws::IAM::Model::CreateAccountAliasOutcome outcome = iam.CreateAccountAlias(
            request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating account alias " << aliasName << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully created account alias " << aliasName <<
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, voir CreateAccountAlias dans le guide de référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour créer un alias de compte

La `create-account-alias` commande suivante crée l'alias `examplecorp` de votre AWS compte.

```
aws iam create-account-alias \
    --account-alias examplecorp
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Votre identifiant de AWS compte et son alias dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir CreateAccountAlias dans AWS CLI la référence des commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.CreateAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
```

```
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAccountAlias {
    public static void main(String[] args) {
        final String usage = """
                Usage:
                    <alias>\s

                Where:
                    alias - The account alias to create (for example,
 myawsaccount).\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        createIAMAccountAlias(iam, alias);
        iam.close();
        System.out.println("Done");
    }

    public static void createIAMAccountAlias(IamClient iam, String alias) {
        try {
            CreateAccountAliasRequest request =
 CreateAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();

            iam.createAccountAlias(request);
            System.out.println("Successfully created account alias: " + alias);

        } catch (IamException e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, voir CreateAccountAlias dans le guide de référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez l'alias de compte.

```
import { CreateAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias - A unique name for the account alias.
 * @returns
 */
export const createAccountAlias = (alias) => {
  const command = new CreateAccountAliasCommand({
    AccountAlias: alias,
  });

  return client.send(command);
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, voir CreateAccountAlias dans le guide de référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, voir CreateAccountAlias dans le guide de référence des AWS SDK for JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun createIAMAccountAlias(alias: String) {
    val request =
        CreateAccountAliasRequest {
            accountAlias = alias
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.createAccountAlias(request)
        println("Successfully created account alias named $alias")
    }
}
```

- Pour plus de détails sur l'API, voir CreateAccountAlias dans le AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell

Exemple 1 : Dans cet exemple, l'alias de votre AWS compte est remplacé par**`mycompanyaws`**. L'adresse de la page de connexion de l'utilisateur devient https://mycompanyaws.signin.aws.amazon.com/console. L'URL d'origine utilisant votre numéro d'identification de compte au lieu de l'alias (https ://<accountidnumber>.signin.aws.amazon.com/console) continue de fonctionner. Cependant, toutes les URL basées sur des alias définies précédemment cessent de fonctionner.

```
New-IAMAccountAlias -AccountAlias mycompanyaws
```

- Pour plus de détails sur l'API, voir CreateAccountAlias dans la référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise
```

- Pour plus de détails sur l'API, voir CreateAccountAlias in AWS SDK for Python (Boto3) API Reference.

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez, créez et supprimez des alias de compte.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info("Account aliases are:")
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info("No account aliases found.")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, voir CreateAccountAlias dans le guide de référence des AWS SDK for Ruby API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreateGroup`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreateGroup`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Créer un groupe et ajouter un utilisateur

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
{ GroupName = groupName });
        return response.Group;
    }
```

- Pour plus de détails sur l'API, voir CreateGroupla section Référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour créer un groupe IAM

La commande `create-group` suivante crée un groupe IAM nommé `Admins`.

```
aws iam create-group \
    --group-name Admins
```

Sortie :

```
{
```

```
    "Group": {
        "Path": "/",
        "CreateDate": "2015-03-09T20:30:24.940Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    }
}
```

Pour plus d'informations, consultez la section Création de groupes d'utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir CreateGroupla section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { CreateGroupCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} groupName
 */
export const createGroup = async (groupName) => {
  const command = new CreateGroupCommand({ GroupName: groupName });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Pour plus de détails sur l'API, voir CreateGroupla section Référence des AWS SDK for JavaScript API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée un nouveau groupe IAM nomméDevelopers.

```
New-IAMGroup -GroupName Developers
```

Sortie :

```
Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 4/14/2015 11:21:31 AM
GroupId    : QNEJ5PM4NFSQCEXAMPLE1
GroupName  : Developers
Path       : /
```

- Pour plus de détails sur l'API, consultez la section CreateGroupRéférence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation `CreateInstanceProfile` avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreateInstanceProfile`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Créer et gérer un service résilient

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /// <summary>
   /// Create a policy, role, and profile that is associated with instances with
a specified name.
   /// An instance's associated profile defines a role that is assumed by the
   /// instance.The role has attached policies that specify the AWS permissions
granted to
   /// clients that run on the instance.
   /// </summary>
   /// <param name="policyName">Name to use for the policy.</param>
   /// <param name="roleName">Name to use for the role.</param>
   /// <param name="profileName">Name to use for the profile.</param>
   /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
   /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
   /// <returns>The Arn of the profile.</returns>
   public async Task<string> CreateInstanceProfileWithName(
       string policyName,
       string roleName,
       string profileName,
       string ssmOnlyPolicyFile,
       List<string>? awsManagedPolicies = null)
   {

       var assumeRoleDoc = "{" +
                                   "\"Version\": \"2012-10-17\"," +
                                   "\"Statement\": [{" +
                                        "\"Effect\": \"Allow\"," +
                                        "\"Principal\": {" +
                                        "\"Service\": [" +
                                            "\"ec2.amazonaws.com\"" +
                                        "]" +
                                        "}," +
```

```
                                "\"Action\": \"sts:AssumeRole\"" +
                                "}]" +
                            "}";

        var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

        var policyArn = "";

        try
        {
            var createPolicyResult = await _amazonIam.CreatePolicyAsync(
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
```

```
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
```

```
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }
```

- Pour plus de détails sur l'API, reportez-vous à la section CreateInstanceProfil dans le AWS SDK for .NET manuel de référence des API.

CLI

AWS CLI

Pour créer un profil d'instance

La commande `create-instance-profile` suivante crée un profil d'instance nommé `Webserver`.

```
aws iam create-instance-profile \
    --instance-profile-name Webserver
```

Sortie :

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AIPAJMBYC7DLSPEXAMPLE",
        "Roles": [],
        "CreateDate": "2015-03-09T20:33:19.626Z",
        "InstanceProfileName": "Webserver",
        "Path": "/",
```

```
        "Arn": "arn:aws:iam::123456789012:instance-profile/Webserver"
    }
}
```

Pour ajouter un rôle à un profil d'instance, utilisez la commande `add-role-to-instance-profile`.

Pour plus d'informations, consultez Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2 dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section CreateInstanceProfil dans AWS CLI la référence des commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
const { InstanceProfile } = await iamClient.send(
  new CreateInstanceProfileCommand({
    InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
  }),
);
await waitUntilInstanceProfileExists(
  { client: iamClient },
  { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
);
```

- Pour plus de détails sur l'API, reportez-vous à la section CreateInstanceProfil dans le AWS SDK for JavaScript manuel de référence des API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée un nouveau profil d'instance IAM nommé**`ProfileForDevEC2Instance`**. Vous devez exécuter la **`Add-IAMRoleToInstanceProfile`** commande séparément pour associer le profil d'instance à un rôle IAM existant qui fournit des autorisations à l'instance. Enfin, attachez le profil d'instance à une instance EC2 lorsque vous la lancez. Pour ce faire, utilisez l'**`New-EC2Instance`**applet de commande avec le paramètre **`InstanceProfile_Arn`** or**`InstanceProfile_Name`**.

```
New-IAMInstanceProfile -InstanceProfileName ProfileForDevEC2Instance
```

Sortie :

```
Arn                 : arn:aws:iam::123456789012:instance-profile/
ProfileForDevEC2Instance
CreateDate          : 4/14/2015 11:31:39 AM
InstanceProfileId   : DYMFXL556EY46EXAMPLE1
InstanceProfileName : ProfileForDevEC2Instance
Path                : /
Roles               : {}
```

- Pour plus de détails sur l'API, consultez la section CreateInstanceProfil dans la référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple crée une politique, un rôle et un profil d'instance et les lie tous ensemble.

```
class AutoScaler:
    """
```

```
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def create_instance_profile(
```

```
        self, policy_file, policy_name, role_name, profile_name,
aws_managed_policies=()
    ):
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        with open(policy_file) as file:
            instance_policy_doc = file.read()

        policy_arn = None
        try:
            pol_response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=instance_policy_doc
            )
```

```
            policy_arn = pol_response["Policy"]["Arn"]
            log.info("Created policy with ARN %s.", policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Policy %s already exists, nothing to do.", policy_name)
                list_pol_response = self.iam_client.list_policies(Scope="Local")
                for pol in list_pol_response["Policies"]:
                    if pol["PolicyName"] == policy_name:
                        policy_arn = pol["Arn"]
                        break
            if policy_arn is None:
                raise AutoScalerError(f"Couldn't create policy {policy_name}:
{err}")

        try:
            self.iam_client.create_role(
                RoleName=role_name,
AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            self.iam_client.attach_role_policy(RoleName=role_name,
PolicyArn=policy_arn)
            for aws_policy in aws_managed_policies:
                self.iam_client.attach_role_policy(
                    RoleName=role_name,
                    PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
                )
            log.info("Created role %s and attached policy %s.", role_name,
policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Role %s already exists, nothing to do.", role_name)
            else:
                raise AutoScalerError(f"Couldn't create role {role_name}: {err}")

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
```

```
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create profile {profile_name} and attach it to
role\n"
                    f"{role_name}: {err}"
                )
        return profile_arn
```

- Pour plus de détails sur l'API, voir CreateInstanceProfile in AWS SDK for Python (Boto3) API Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreateLoginProfile`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreateLoginProfile`.

CLI

AWS CLI

Pour créer un mot de passe pour un utilisateur IAM

Pour créer un mot de passe pour un utilisateur IAM, nous vous recommandons d'utiliser le `--cli-input-json` paramètre pour transmettre un fichier JSON contenant le mot de

passe. Cette méthode vous permet de créer un mot de passe fort avec des caractères non alphanumériques. Il peut être difficile de créer un mot de passe contenant des caractères non alphanumériques lorsque vous le transmettez en tant que paramètre de ligne de commande.

Pour utiliser le `--cli-input-json` paramètre, commencez par utiliser la `create-login-profile` commande avec le `--generate-cli-skeleton` paramètre, comme dans l'exemple suivant.

```
aws iam create-login-profile \
    --generate-cli-skeleton > create-login-profile.json
```

La commande précédente crée un fichier JSON appelé create-login-profile .json que vous pouvez utiliser pour renseigner les informations d'une `create-login-profile` commande suivante. Par exemple :

```
{
    "UserName": "Bob",
    "Password": "&1-3a6u:RA0djs",
    "PasswordResetRequired": true
}
```

Ensuite, pour créer un mot de passe pour un utilisateur IAM, réutilisez la `create-login-profile` commande, en passant cette fois le `--cli-input-json` paramètre pour spécifier votre fichier JSON. La `create-login-profile` commande suivante utilise le `--cli-input-json` paramètre avec un fichier JSON appelé create-login-profile .json.

```
aws iam create-login-profile \
    --cli-input-json file://create-login-profile.json
```

Sortie :

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2015-03-10T20:55:40.274Z",
        "PasswordResetRequired": true
    }
}
```

Si le nouveau mot de passe enfreint la politique de mot de passe du compte, la commande renvoie une `PasswordPolicyViolation` erreur.

Pour modifier le mot de passe d'un utilisateur qui en possède déjà un, utilisez`update-login-profile`. Pour définir une politique de mot de passe pour le compte, utilisez la `update-account-password-policy` commande.

Si la politique de mot de passe du compte le permet, les utilisateurs IAM peuvent modifier leurs propres mots de passe à l'aide de la `change-password` commande.

Pour plus d'informations, consultez la section Gestion des mots de passe pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section CreateLoginProfil dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée un mot de passe (temporaire) pour l'utilisateur IAM nommé Bob et définit l'indicateur qui oblige l'utilisateur à modifier le mot de passe lors de sa prochaine **Bob** connexion.

```
New-IAMLoginProfile -UserName Bob -Password P@ssw0rd -PasswordResetRequired $true
```

Sortie :

```
CreateDate                PasswordResetRequired             UserName
----------                ---------------------             --------
4/14/2015 12:26:30 PM     True                              Bob
```

- Pour plus de détails sur l'API, consultez la section CreateLoginProfil dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreateOpenIdConnectProvider`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreateOpenIdConnectProvider`.

CLI

AWS CLI

Pour créer un fournisseur OpenID Connect (OIDC)

Pour créer un fournisseur OpenID Connect (OIDC), nous vous recommandons d'utiliser le `--cli-input-json` paramètre pour transmettre un fichier JSON contenant les paramètres requis. Lorsque vous créez un fournisseur OIDC, vous devez transmettre l'URL du fournisseur, qui doit commencer `https://` par. Il peut être difficile de transmettre l'URL en tant que paramètre de ligne de commande, car les deux points (:) et les barres obliques (/) ont une signification particulière dans certains environnements de ligne de commande. L'utilisation du `--cli-input-json` paramètre permet de contourner cette limitation.

Pour utiliser le `--cli-input-json` paramètre, commencez par utiliser la `create-open-id-connect-provider` commande avec le `--generate-cli-skeleton` paramètre, comme dans l'exemple suivant.

```
aws iam create-open-id-connect-provider \
    --generate-cli-skeleton > create-open-id-connect-provider.json
```

La commande précédente crée un fichier JSON appelé create-open-id-connect -provider.json que vous pouvez utiliser pour renseigner les informations d'une commande suivante. `create-open-id-connect-provider` Par exemple :

```
{
    "Url": "https://server.example.com",
    "ClientIDList": [
        "example-application-ID"
    ],
    "ThumbprintList": [
        "c3768084dfb3d2b68b7897bf5f565da8eEXAMPLE"
    ]
}
```

Ensuite, pour créer le fournisseur OpenID Connect (OIDC), réutilisez la `create-open-id-connect-provider` commande, en passant cette fois le `--cli-input-json` paramètre

pour spécifier votre fichier JSON. La `create-open-id-connect-provider` commande suivante utilise le `--cli-input-json` paramètre avec un fichier JSON appelé create-open-id-connect -provider.json.

```
aws iam create-open-id-connect-provider \
    --cli-input-json file://create-open-id-connect-provider.json
```

Sortie :

```
{
    "OpenIDConnectProviderArn": "arn:aws:iam::123456789012:oidc-provider/
server.example.com"
}
```

Pour plus d'informations sur les fournisseurs OIDC, consultez la section Création de fournisseurs d'identité OpenID Connect (OIDC)AWS dans le guide de l'utilisateur IAM.

Pour plus d'informations sur l'obtention d'empreintes digitales pour un fournisseur OIDC, consultez la section Obtention de l'empreinte numérique pour un fournisseur d'identité OpenID Connect dans le guide de l'utilisateur IAM.AWS

- Pour plus de détails sur l'API, voir CreateOpenIdConnectProvider dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée un fournisseur IAM OIDC associé au service de fournisseur compatible OIDC figurant sur l'URL **`https://example.oidcprovider.com`** et l'ID client. **`my-testapp-1`** Le fournisseur OIDC fournit l'empreinte numérique. Pour authentifier l'empreinte numérique, suivez les étapes indiquées sur http://docs.aws.amazon.com/IAM/latest/ /identity-providers-oidc-obtain-thumbprint.html. UserGuide

```
New-IAMOpenIDConnectProvider -Url https://example.oidcprovider.com -ClientIDList
 my-testapp-1 -ThumbprintList 990F419EXAMPLEECF12DDEDA5EXAMPLE52F20D9E
```

Sortie :

```
arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Pour plus de détails sur l'API, consultez la section CreateOpenIdConnectFournisseur dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreatePolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreatePolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Créer un groupe et ajouter un utilisateur
- Créer un utilisateur et assumer d'un rôle
- Créer des utilisateurs en lecture seule et en lecture-écriture
- Gérer les politiques
- Utiliser l'API IAM Policy Builder

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
```

```
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK for .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
```

```
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
```

```
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
Aws::String AwsDoc::IAM::createPolicy(const Aws::String &policyName,
                                      const Aws::String &rsrcArn,
```

```
                                        const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreatePolicyRequest request;
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(BuildSamplePolicyDocument(rsrcArn));

    Aws::IAM::Model::CreatePolicyOutcome outcome = iam.CreatePolicy(request);
    Aws::String result;
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating policy " << policyName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        result = outcome.GetResult().GetPolicy().GetArn();
        std::cout << "Successfully created policy " << policyName <<
                  std::endl;
    }

    return result;
}

Aws::String AwsDoc::IAM::BuildSamplePolicyDocument(const Aws::String &rsrc_arn) {
    std::stringstream stringStream;
    stringStream << "{"
                 << "  \"Version\": \"2012-10-17\","
                 << "  \"Statement\": ["
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": \"logs:CreateLogGroup\","
                 << "        \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    },"
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": ["
                 << "            \"dynamodb:DeleteItem\","
                 << "            \"dynamodb:GetItem\","
                 << "            \"dynamodb:PutItem\","
                 << "            \"dynamodb:Scan\","
                 << "            \"dynamodb:UpdateItem\""
                 << "       ],"
```

```
                << "       \"Resource\": \""
                << rsrc_arn
                << "\""
                << "    }"
                << "   ]"
                << "}";

    return stringStream.str();
}
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK for C++ API.

CLI

AWS CLI

Exemple 1 : Pour créer une politique gérée par le client

La commande suivante crée une politique gérée par le client nommée `my-policy`.

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy
```

Le fichier `policy` est un document JSON dans le dossier actuel qui accorde un accès en lecture seule au dossier `shared` dans un compartiment Amazon S3 nommé `my-bucket`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket/shared/*"
            ]
```

```
        }
    ]
}
```

Sortie :

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "CreateDate": "2015-06-01T19:31:18.620Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "ZXR6A36LTYANPAI7NJ5UV",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::0123456789012:policy/my-policy",
        "UpdateDate": "2015-06-01T19:31:18.620Z"
    }
}
```

Pour plus d'informations sur l'utilisation de fichiers comme entrée pour les paramètres de chaîne, voir Spécifier les valeurs des paramètres pour la AWS CLI dans le guide de l'utilisateur de la AWS CLI.

Exemple 2 : Pour créer une politique gérée par le client avec une description

La commande suivante crée une politique gérée par le client nommée `my-policy` avec une description immuable :

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --description "This policy grants access to all Put, Get, and List actions
 for my-bucket"
```

Le fichier `policy.json` est un document JSON dans le dossier actuel qui accorde un accès à toutes les actions Pull, List et Get dans un compartiment Amazon S3 nommé `my-bucket`.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                 "s3:ListBucket*",
                 "s3:PutBucket*",
                 "s3:GetBucket*"
             ],
             "Resource": [
                 "arn:aws:s3:::my-bucket"
             ]
         }
     ]
 }
```

Sortie :

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T22:38:47+00:00",
        "UpdateDate": "2023-05-24T22:38:47+00:00"
    }
}
```

Pour en savoir plus sur les politiques basées sur l'identité, consultez Politiques basées sur l'identité et Politiques basées sur une ressource dans le Guide de l'utilisateur AWS IAM.

Exemple 3 : Pour créer une politique gérée par le client avec des balises

La commande suivante crée une politique gérée par le client nommée `my-policy` avec des balises. Cet exemple utilise l'indicateur de paramètre `--tags` avec les balises au format JSON suivantes : `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. L'indicateur `--tags` peut également être utilisé avec des balises au format raccourci : `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Le fichier `policy.json` est un document JSON dans le dossier actuel qui accorde un accès à toutes les actions Pull, List et Get dans un compartiment Amazon S3 nommé `my-bucket`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                 "s3:ListBucket*",
                 "s3:PutBucket*",
                 "s3:GetBucket*"
             ],
             "Resource": [
                 "arn:aws:s3:::my-bucket"
             ]
         }
     ]
 }
```

Sortie :

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::12345678012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T23:16:39+00:00",
        "UpdateDate": "2023-05-24T23:16:39+00:00",
        "Tags": [
            {
```

```
                "Key": "Department",
                "Value": "Accounting"
            },
                "Key": "Location",
                "Value": "Seattle"
            {

        ]
    }
}
```

Pour plus d'informations sur les politiques de balisage, consultez Balisage des politiques gérées par le client dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
```

```
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(policyName string, actions []string,
  resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(context.TODO(),
 &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK for Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreatePolicyRequest;
import software.amazon.awssdk.services.iam.model.CreatePolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyRequest;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreatePolicy {

    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"dynamodb:DeleteItem\"," +
            "            \"dynamodb:GetItem\"," +
            "            \"dynamodb:PutItem\"," +
            "            \"dynamodb:Scan\"," +
            "            \"dynamodb:UpdateItem\"" +
```

```
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static void main(String[] args) {

        final String usage = """
                Usage:
                    CreatePolicy <policyName>\s

                Where:
                    policyName - A unique policy name.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String policyName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMPolicy(iam, policyName);
        System.out.println("Successfully created a policy with this ARN value: "
+ result);
        iam.close();
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument)
                    .build();

            CreatePolicyResponse response = iam.createPolicy(request);
```

```
            // Wait until the policy is created.
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
 iamWaiter.waitUntilPolicyExists(polRequest);

 waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez la politique.

```
import { CreatePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
```

```
 * @param {string} policyName
 */
export const createPolicy = (policyName) => {
  const command = new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: "*",
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  });

  return client.send(command);
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });
```

```
var myManagedPolicy = {
  Version: "2012-10-17",
  Statement: [
    {
      Effect: "Allow",
      Action: "logs:CreateLogGroup",
      Resource: "RESOURCE_ARN",
    },
    {
      Effect: "Allow",
      Action: [
        "dynamodb:DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:PutItem",
        "dynamodb:Scan",
        "dynamodb:UpdateItem",
      ],
      Resource: "RESOURCE_ARN",
    },
  ],
};

var params = {
  PolicyDocument: JSON.stringify(myManagedPolicy),
  PolicyName: "myDynamoDBPolicy",
};

iam.createPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK for JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun createIAMPolicy(policyNameVal: String?): String {
    val policyDocumentVal =
        "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"dynamodb:DeleteItem\"," +
            "            \"dynamodb:GetItem\"," +
            "            \"dynamodb:PutItem\"," +
            "            \"dynamodb:Scan\"," +
            "            \"dynamodb:UpdateItem\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}"

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

    public function createPolicy(string $policyName, string $policyDocument)
    {
        $result = $this->customWaiter(function () use ($policyName,
 $policyDocument) {
            return $this->iamClient->createPolicy([
                'PolicyName' => $policyName,
                'PolicyDocument' => $policyDocument,
            ]);
        });
        return $result['Policy'];
    }
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK for PHP API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée une nouvelle stratégie IAM dans le AWS compte courant nommée Le fichier **`MySamplePolicy.json`** fournit **`MySamplePolicy`** le contenu de la stratégie. Notez que vous devez utiliser le paramètre **`-Raw`** switch pour traiter correctement le fichier de politique JSON.

```
New-IAMPolicy -PolicyName MySamplePolicy -PolicyDocument (Get-Content -Raw
 MySamplePolicy.json)
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 4/14/2015 2:45:59 PM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : LD4KP6HVFE7WGEXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 4/14/2015 2:45:59 PM
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::my-bucket/*' to allow actions on all
 objects
                         in the bucket named 'my-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy
```

- Pour plus de détails sur l'API, consultez CreatePolicyle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, attache et détache les politiques de rôle.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
```

```
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn create_policy(
    client: &iamClient,
    policy_name: &str,
    policy_document: &str,
) -> Result<Policy, iamError> {
    let policy = client
        .create_policy()
        .policy_name(policy_name)
        .policy_document(policy_document)
        .send()
        .await?;
```

```
    Ok(policy.policy.unwrap())
}
```

- Pour plus de détails sur l'API, voir CreatePolicyla section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    public func createPolicy(name: String, policyDocument: String) async throws -
> IAMClientTypes.Policy {
        let input = CreatePolicyInput(
            policyDocument: policyDocument,
            policyName: name
        )
        do {
            let output = try await iamClient.createPolicy(input: input)
            guard let policy = output.policy else {
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
        } catch {
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous CreatePolicyà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreatePolicyVersion`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreatePolicyVersion`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer les politiques

CLI

AWS CLI

Pour créer une nouvelle version de la politique gérée

Cet exemple crée une nouvelle version `v2` de la politique IAM dont l'ARN est `arn:aws:iam::123456789012:policy/MyPolicy` et en fait la version par défaut.

```
aws iam create-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --policy-document file://NewPolicyVersion.json \
    --set-as-default
```

Sortie :

```
{
    "PolicyVersion": {
        "CreateDate": "2015-06-16T18:56:03.721Z",
        "VersionId": "v2",
        "IsDefaultVersion": true
    }
}
```

Pour plus d'informations, consultez Gestion des versions des politiques IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir CreatePolicyVersion dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée une nouvelle version « v2 » de la politique IAM dont l'ARN est l'ARN **`arn:aws:iam::123456789012:policy/MyPolicy`** et en fait la version par défaut. Le **`NewPolicyVersion.json`** fichier fournit le contenu de la politique. Notez que vous devez utiliser le paramètre **`-Raw`** switch pour traiter correctement le fichier de politique JSON.

```
New-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy -
PolicyDocument (Get-content -Raw NewPolicyVersion.json) -SetAsDefault $true
```

Sortie :

```
CreateDate                          Document                 IsDefaultVersion
          VersionId
----------                          --------                 ----------------
          ---------
4/15/2015 10:54:54 AM                                        True
          v2
```

- Pour plus de détails sur l'API, consultez CreatePolicyla section Version dans la référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
        return policy_version
```

- Pour plus de détails sur l'API, voir CreatePolicyVersion in AWS SDK for Python (Boto3) API Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreateRole`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreateRole`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Créer un groupe et ajouter un utilisateur
- Créer un utilisateur et assumer d'un rôle
- Gérer les rôles

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /// <summary>
   /// Create a new IAM role.
   /// </summary>
   /// <param name="roleName">The name of the IAM role.</param>
   /// <param name="rolePolicyDocument">The name of the IAM policy document
   /// for the new role.</param>
   /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
   public async Task<string> CreateRoleAsync(string roleName, string
rolePolicyDocument)
   {
       var request = new CreateRoleRequest
       {
           RoleName = roleName,
           AssumeRolePolicyDocument = rolePolicyDocument,
```

```
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS SDK for .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
```

```
#     And:
#       0 - If successful.
#       1 - If it fails.
################################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
```

```
    return 1
  fi

  response=$(aws iam create-role \
    --role-name "$role_name" \
    --assume-role-policy-document "$policy_document" \
    --output text \
    --query Role.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::createIamRole(
        const Aws::String &roleName,
        const Aws::String &policy,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient client(clientConfig);
```

```
    Aws::IAM::Model::CreateRoleRequest request;

    request.SetRoleName(roleName);
    request.SetAssumeRolePolicyDocument(policy);

    Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const Aws::IAM::Model::Role iamRole = outcome.GetResult().GetRole();
        std::cout << "Created role " << iamRole.GetRoleName() << "\n";
        std::cout << "ID: " << iamRole.GetRoleId() << "\n";
        std::cout << "ARN: " << iamRole.GetArn() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS SDK for C++ API.

CLI

AWS CLI

Exemple 1 : Pour créer un rôle IAM

La commande `create-role` suivante crée un rôle nommé `Test-Role` et lui attache une politique d'approbation.

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json
```

Sortie :

```
{
    "Role": {
        "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
```

```
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "CreateDate": "2013-06-07T20:43:32.821Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

La politique d'approbation est définie sous la forme d'un document JSON dans le fichier Test-Role-Trust-Policy.json. (Le nom et l'extension du fichier n'ont aucune importance.) La politique d'approbation doit spécifier un principal.

Pour attacher une politique des autorisations à un rôle, utilisez la commande `put-role-policy`.

Pour plus d'informations, consultez Création de rôles IAM dans le Guide de l'utilisateur AWS IAM.

Exemple 2 : Pour créer un rôle IAM avec une durée de session maximale spécifiée

La commande `create-role` suivante crée un rôle nommé `Test-Role` et définit une durée de session maximale de 7 200 secondes (2 heures).

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --max-session-duration 7200
```

Sortie :

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:role/Test-Role",
        "CreateDate": "2023-05-24T23:50:25+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
```

```
                    "Principal": {
                        "AWS": "arn:aws:iam::12345678012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        }
    }
}
```

Pour plus d'informations, consultez la section Modification de la durée maximale de session (AWS API) d'un rôle dans le guide de l'utilisateur AWS IAM.

Exemple 3 : Pour créer un rôle IAM avec des balises

La commande suivante crée un rôle IAM `Test-Role` avec des balises. Cet exemple utilise l'indicateur de paramètre `--tags` avec les balises au format JSON suivantes : `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. L'indicateur `--tags` peut également être utilisé avec des balises au format raccourci : `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Sortie :

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role",
        "CreateDate": "2023-05-25T23:29:41+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
```

```
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        },
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

Pour plus d'informations, veuillez consulter Étiquette de rôles IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
```

```
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(roleName string, trustedUserArn string)
 (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action: []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(context.TODO(),
 &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS SDK for Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;
import software.amazon.awssdk.services.iam.model.CreateRoleRequest;
import software.amazon.awssdk.services.iam.model.CreateRoleResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import java.io.FileReader;

/*
*   This example requires a trust policy document. For more information, see:
*   https://aws.amazon.com/blogs/security/how-to-use-trust-policies-with-iam-
roles/
*
*
*  In addition, set up your development environment, including your credentials.
*
*  For information, see this documentation topic:
*
*  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class CreateRole {
    public static void main(String[] args) throws Exception {
        final String usage = """
                Usage:
                    <rolename> <fileLocation>\s
```

```
                Where:
                    rolename - The name of the role to create.\s
                    fileLocation - The location of the JSON document that
represents the trust policy.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String rolename = args[0];
        String fileLocation = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMRole(iam, rolename, fileLocation);
        System.out.println("Successfully created user: " + result);
        iam.close();
    }

    public static String createIAMRole(IamClient iam, String rolename, String
fileLocation) throws Exception {
        try {
            JSONObject jsonObject = (JSONObject)
readJsonSimpleDemo(fileLocation);
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(jsonObject.toJSONString())
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
```

```
    }

    public static Object readJsonSimpleDemo(String filename) throws Exception {
        FileReader reader = new FileReader(filename);
        JSONParser jsonParser = new JSONParser();
        return jsonParser.parse(reader);
    }
}
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez le rôle.

```
import { CreateRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const createRole = (roleName) => {
  const command = new CreateRoleCommand({
    AssumeRolePolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Principal: {
            Service: "lambda.amazonaws.com",
```

```
          },
          Action: "sts:AssumeRole",
        },
      ],
    }),
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS SDK for JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

    /**
```

```
     * @param string $roleName
     * @param string $rolePolicyDocument
     * @return array
     * @throws AwsException
     */
    public function createRole(string $roleName, string $rolePolicyDocument)
    {
        $result = $this->customWaiter(function () use ($roleName,
 $rolePolicyDocument) {
            return $this->iamClient->createRole([
                'AssumeRolePolicyDocument' => $rolePolicyDocument,
                'RoleName' => $roleName,
            ]);
        });
        return $result['Role'];
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS SDK for PHP API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée un nouveau rôle nommé **MyNewRole** et y attache la politique trouvée dans le fichier**NewRoleTrustPolicy.json**. Notez que vous devez utiliser le paramètre **-Raw** switch pour traiter correctement le fichier de politique JSON. Le document de politique affiché dans la sortie est codé en URL. Il est décodé dans cet exemple à l'aide de la méthode **UrlDecode** .NET.

```
$results = New-IAMRole -AssumeRolePolicyDocument (Get-Content -raw
 NewRoleTrustPolicy.json) -RoleName MyNewRole
$results
```

Sortie :

```
Arn                      : arn:aws:iam::123456789012:role/MyNewRole
AssumeRolePolicyDocument : %7B%0D%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C
%0D%0A%20%20%22Statement%22
```

```
                            %3A%20%5B%0D%0A%20%20%20%20%7B%0D%0A
%20%20%20%20%20%20%22Sid%22%3A%20%22%22%2C
                            %0D%0A%20%20%20%20%20%20%22Effect%22%3A%20%22Allow
%22%2C%0D%0A%20%20%20%20%20%20
                            %22Principal%22%3A%20%7B%0D%0A
%20%20%20%20%20%20%20%20%22AWS%22%3A%20%22arn%3Aaws
                            %3Aiam%3A%3A123456789012%3ADavid%22%0D%0A
%20%20%20%20%20%20%7D%2C%0D%0A%20%20%20
                            %20%20%20%22Action%22%3A%20%22sts%3AAssumeRole%22%0D
%0A%20%20%20%20%7D%0D%0A%20
                            %20%5D%0D%0A%7D
CreateDate               : 4/15/2015 11:04:23 AM
Path                     : /
RoleId                   : V5PAJI2KPN4EAEXAMPLE1
RoleName                 : MyNewRole

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:David"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
    """
    trust_policy = {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role
```

- Pour plus de détails sur l'API, consultez CreateRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Creates a role and attaches policies to it.
 #
 # @param role_name [String] The name of the role.
 # @param assume_role_policy_document [Hash] The trust relationship policy
document.
 # @param policy_arns [Array<String>] The ARNs of the policies to attach.
 # @return [String, nil] The ARN of the new role if successful, or nil if an
error occurred.
 def create_role(role_name, assume_role_policy_document, policy_arns)
   response = @iam_client.create_role(
     role_name: role_name,
     assume_role_policy_document: assume_role_policy_document.to_json
   )
   role_arn = response.role.arn

   policy_arns.each do |policy_arn|
     @iam_client.attach_role_policy(
       role_name: role_name,
       policy_arn: policy_arn
     )
   end

   role_arn
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error creating role: #{e.message}")
   nil
```

```
  end
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn create_role(
    client: &iamClient,
    role_name: &str,
    role_policy_document: &str,
) -> Result<Role, iamError> {
    let response: CreateRoleOutput = loop {
        if let Ok(response) = client
            .create_role()
            .role_name(role_name)
            .assume_role_policy_document(role_policy_document)
            .send()
            .await
        {
            break response;
        }
    };

    Ok(response.role.unwrap())
}
```

- Pour plus de détails sur l'API, voir CreateRolela section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   public func createRole(name: String, policyDocument: String) async throws ->
String {
       let input = CreateRoleInput(
           assumeRolePolicyDocument: policyDocument,
           roleName: name
       )
       do {
           let output = try await client.createRole(input: input)
           guard let role = output.role else {
               throw ServiceHandlerError.noSuchRole
           }
           guard let id = role.roleId else {
               throw ServiceHandlerError.noSuchRole
           }
           return id
       } catch {
           throw error
       }
   }
```

- Pour plus de détails sur l'API, reportez-vous CreateRoleà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreateSAMLProvider`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreateSAMLProvider`.

CLI

AWS CLI

Pour créer un fournisseur SAML

Cet exemple crée un nouveau fournisseur SAML dans IAM nommé `MySAMLProvider`. Il est décrit par le document de métadonnées SAML présent dans le fichier `SAMLMetaData.xml`.

```
aws iam create-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --name MySAMLProvider
```

Sortie :

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/MySAMLProvider"
}
```

Pour plus d'informations, consultez Création de fournisseurs d'identité SAML IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus d'informations sur l'API, veuillez consulter la rubrique CreateSAMLProvider dans la Référence des commandes AWS CLI .

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { CreateSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "fs";
import * as path from "path";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";

const client = new IAMClient({});

/**
 * This sample document was generated using Auth0.
 * For more information on generating this document,
 see https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_providers_create_saml.html#samlstep1.
 */
const sampleMetadataDocument = readFileSync(
  path.join(
    dirnameFromMetaUrl(import.meta.url),
    "../../../../resources/sample_files/sample_saml_metadata.xml",
  ),
);

/**
 *
 * @param {*} providerName
 * @returns
 */
export const createSAMLProvider = async (providerName) => {
  const command = new CreateSAMLProviderCommand({
    Name: providerName,
    SAMLMetadataDocument: sampleMetadataDocument.toString(),
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Pour les détails de l'API, consultez CreateSAMLProvider dans la Référence de l'API AWS SDK for JavaScript .

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée une nouvelle entité de fournisseur SAML dans IAM. Il est nommé **`MySAMLProvider`** et décrit par le document de métadonnées SAML trouvé dans le fichier**`SAMLMetaData.xml`**, qui a été téléchargé séparément depuis le site Web du fournisseur de services SAML.

```
New-IAMSAMLProvider -Name MySAMLProvider -SAMLMetadataDocument (Get-Content -Raw
 SAMLMetaData.xml)
```

Sortie :

```
arn:aws:iam::123456789012:saml-provider/MySAMLProvider
```

- Pour plus de détails sur l'API, consultez CreateSAMLProvider dans la référence des applets de commande.AWS Tools for PowerShell

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreateServiceLinkedRole`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreateServiceLinkedRole`.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
```

```
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section Référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour créer un rôle lié à un service

L'`create-service-linked-role`exemple suivant crée un rôle lié à un service pour le AWS service spécifié et y joint la description spécifiée.

```
aws iam create-service-linked-role \
    --aws-service-name lex.amazonaws.com \
    --description "My service-linked role to support Lex"
```

Sortie :

```
{
    "Role": {
        "Path": "/aws-service-role/lex.amazonaws.com/",
        "RoleName": "AWSServiceRoleForLexBots",
        "RoleId": "AROA1234567890EXAMPLE",
```

```
        "Arn": "arn:aws:iam::1234567890:role/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots",
        "CreateDate": "2019-04-17T20:34:14+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": [
                        "sts:AssumeRole"
                    ],
                    "Effect": "Allow",
                    "Principal": {
                        "Service": [
                            "lex.amazonaws.com"
                        ]
                    }
                }
            ]
        }
    }
}
```

Pour plus d'informations, consultez Utilisation des rôles liés à un service dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
```

```
type RoleWrapper struct {
 IamClient *iam.Client
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(serviceName string,
 description string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section Référence des AWS SDK for Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez un rôle lié à un service.

```
import {
  CreateServiceLinkedRoleCommand,
  GetRoleCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} serviceName
 */
export const createServiceLinkedRole = async (serviceName) => {
  const command = new CreateServiceLinkedRoleCommand({
    // For a list of AWS services that support service-linked roles,
    // see https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_aws-
services-that-work-with-iam.html.
    //
    // For a list of AWS service endpoints, see https://docs.aws.amazon.com/
general/latest/gr/aws-service-information.html.
    AWSServiceName: serviceName,
  });
  try {
    const response = await client.send(command);
    console.log(response);
    return response;
  } catch (caught) {
    if (
      caught instanceof Error &&
      caught.name === "InvalidInputException" &&
      caught.message.includes(
        "Service role name AWSServiceRoleForElasticBeanstalk has been taken in
 this account",
      )
    ) {
      console.warn(caught.message);
      return client.send(
        new GetRoleCommand({ RoleName: "AWSServiceRoleForElasticBeanstalk" }),
      );
    } else {
      throw caught;
    }
  }
```

```
};
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section Référence des AWS SDK for JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function createServiceLinkedRole($awsServiceName, $customSuffix = "",
 $description = "")
    {
        $createServiceLinkedRoleArguments = ['AWSServiceName' =>
 $awsServiceName];
        if ($customSuffix) {
            $createServiceLinkedRoleArguments['CustomSuffix'] = $customSuffix;
        }
        if ($description) {
            $createServiceLinkedRoleArguments['Description'] = $description;
        }
        return $this->iamClient-
>createServiceLinkedRole($createServiceLinkedRoleArguments);
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section Référence des AWS SDK for PHP API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée un rôle lié à un service pour le service de dimensionnement automatique.

```
New-IAMServiceLinkedRole -AWSServiceName autoscaling.amazonaws.com -CustomSuffix
 RoleNameEndsWithThis -Description "My service-linked role to support
 autoscaling"
```

- Pour plus de détails sur l'API, consultez la section CreateServiceLinkedRoleRéférence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def create_service_linked_role(service_name, description):
    """
    Creates a service-linked role.

    :param service_name: The name of the service that owns the role.
    :param description: A description to give the role.
    :return: The newly created role.
    """
    try:
        response = iam.meta.client.create_service_linked_role(
            AWSServiceName=service_name, Description=description
        )
        role = iam.Role(response["Role"]["RoleName"])
        logger.info("Created service-linked role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create service-linked role for %s.",
 service_name)
```

```
        raise
    else:
        return role
```

- Pour plus de détails sur l'API, consultez CreateServiceLinkedRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Creates a service-linked role
 #
 # @param service_name [String] The service name to create the role for.
 # @param description [String] The description of the service-linked role.
 # @param suffix [String] Suffix for customizing role name.
 # @return [String] The name of the created role
 def create_service_linked_role(service_name, description, suffix)
   response = @iam_client.create_service_linked_role(
     aws_service_name: service_name, description: description, custom_suffix:
suffix,)
   role_name = response.role.role_name
   @logger.info("Created service-linked role #{role_name}.")
   role_name
 rescue Aws::Errors::ServiceError => e
   @logger.error("Couldn't create service-linked role for #{service_name}.
Here's why:")
   @logger.error("\t#{e.code}: #{e.message}")
   raise
 end
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn create_service_linked_role(
    client: &iamClient,
    aws_service_name: String,
    custom_suffix: Option<String>,
    description: Option<String>,
) -> Result<CreateServiceLinkedRoleOutput,
 SdkError<CreateServiceLinkedRoleError>> {
    let response = client
        .create_service_linked_role()
        .aws_service_name(aws_service_name)
        .set_custom_suffix(custom_suffix)
        .set_description(description)
        .send()
        .await?;

    Ok(response)
}
```

- Pour plus de détails sur l'API, voir CreateServiceLinkedRolela section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   public func createServiceLinkedRole(service: String, suffix: String? = nil,
description: String?)
                   async throws -> IAMClientTypes.Role {
       let input = CreateServiceLinkedRoleInput(
           awsServiceName: service,
           customSuffix: suffix,
           description: description
       )
       do {
           let output = try await client.createServiceLinkedRole(input: input)
           guard let role = output.role else {
               throw ServiceHandlerError.noSuchRole
           }
           return role
       } catch {
           throw error
       }
   }
```

- Pour plus de détails sur l'API, reportez-vous CreateServiceLinkedRoleà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreateUser`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreateUser`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Créer un groupe et ajouter un utilisateur
- Créer un utilisateur et assumer d'un rôle
- Créer des utilisateurs en lecture seule et en lecture-écriture

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK for .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
```

```
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi
```

```
  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateUserRequest create_request;
    create_request.SetUserName(userName);

    auto create_outcome = iam.CreateUser(create_request);
    if (!create_outcome.IsSuccess()) {
        std::cerr << "Error creating IAM user " << userName << ":" <<
                  create_outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully created IAM user " << userName << std::endl;
    }

    return create_outcome.IsSuccess();
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK for C++ API.

CLI

AWS CLI

Exemple 1 : Pour créer un utilisateur IAM

La commande `create-user` suivante crée un utilisateur IAM nommé Bob dans le compte actuel.

```
aws iam create-user \
    --user-name Bob
```

Sortie :

```
{
    "User": {
        "UserName": "Bob",
        "Path": "/",
        "CreateDate": "2023-06-08T03:20:41.270Z",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Bob"
```

```
    }
}
```

Pour plus d'informations, consultez la section Création d'un utilisateur IAM dans votre AWS compte dans le guide de l'utilisateur AWS IAM.

Exemple 2 : Pour créer un utilisateur IAM à un chemin spécifié

La commande `create-user` suivante crée un utilisateur IAM nommé Bob au chemin spécifié.

```
aws iam create-user \
    --user-name Bob \
    --path /division_abc/subdivision_xyz/
```

Sortie :

```
{
    "User": {
        "Path": "/division_abc/subdivision_xyz/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/division_abc/subdivision_xyz/Bob",
        "CreateDate": "2023-05-24T18:20:17+00:00"
    }
}
```

Pour plus d'informations, consultez Identifiants IAM dans le Guide de l'utilisateur AWS IAM.

Exemple 3 : Pour créer un utilisateur IAM avec des balises

La commande `create-user` suivante crée un utilisateur IAM nommé Bob avec des balises. Cet exemple utilise l'indicateur de paramètre `--tags` avec les balises au format JSON suivantes : `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. L'indicateur `--tags` peut également être utilisé avec des balises au format raccourci : `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

```
aws iam create-user \
    --user-name Bob \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Sortie :

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-25T17:14:21+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

Pour plus d'informations, consultez Étiquette d'utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

Exemple 3 : Pour créer un utilisateur IAM avec une limite des autorisations définie

La `create-user` commande suivante crée un utilisateur IAM nommé Bob avec la limite d'autorisations d'FullAccessAmazonS3.

```
aws iam create-user \
    --user-name Bob \
    --permissions-boundary arn:aws:iam::aws:policy/AmazonS3FullAccess
```

Sortie :

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-24T17:50:53+00:00",
```

```
        "PermissionsBoundary": {
        "PermissionsBoundaryType": "Policy",
        "PermissionsBoundaryArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
        }
    }
}
```

Pour plus d'informations, consultez Limites d'autorisations pour les entités IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(context.TODO(),
 &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
```

```
  user = result.User
 }
 return user, err
}
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK for Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreateUserRequest;
import software.amazon.awssdk.services.iam.model.CreateUserResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;
import software.amazon.awssdk.services.iam.model.GetUserRequest;
import software.amazon.awssdk.services.iam.model.GetUserResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateUser {
    public static void main(String[] args) {
```

```
        final String usage = """

                Usage:
                    <username>\s

                Where:
                    username - The name of the user to create.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMUser(iam, username);
        System.out.println("Successfully created user: " + result);
        iam.close();
    }

    public static String createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            CreateUserResponse response = iam.createUser(request);

            // Wait until the user is created.
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);
```

```
 waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user().userName();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez l'utilisateur.

```
import { CreateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const createUser = (name) => {
  const command = new CreateUserCommand({ UserName: name });
  return client.send(command);
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    iam.createUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
    });
  } else {
    console.log(
      "User " + process.argv[2] + " already exists",
      data.User.UserId
    );
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK for JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun createIAMUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}
```

- Pour plus de détails sur l'API, consultez CreateUserla section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";


    /**
     * @param string $name
     * @return array
     * @throws AwsException
     */
    public function createUser(string $name): array
    {
        $result = $this->iamClient->createUser([
            'UserName' => $name,
        ]);

        return $result['User'];
    }
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK for PHP API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée un utilisateur IAM nommé**Bob**. Si Bob doit se connecter à la AWS console, vous devez exécuter la commande séparément **New-IAMLoginProfile** pour

créer un profil de connexion avec un mot de passe. Si Bob doit exécuter AWS PowerShell des commandes CLI multiplateformes ou effectuer des appels d' AWS API, vous devez exécuter la **`New-IAMAccessKey`** commande séparément pour créer des clés d'accès.

```
New-IAMUser -UserName Bob
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 4/22/2015 12:02:11 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : AIDAJWGEFDMEMEXAMPLE1
UserName         : Bob
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
```

```
            raise
        else:
            return user
```

- Pour plus de détails sur l'API, consultez CreateUserle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Creates a user and their login profile
 #
 # @param user_name [String] The name of the user
 # @param initial_password [String] The initial password for the user
 # @return [String, nil] The ID of the user if created, or nil if an error
occurred
 def create_user(user_name, initial_password)
   response = @iam_client.create_user(user_name: user_name)
   @iam_client.wait_until(:user_exists, user_name: user_name)
   @iam_client.create_login_profile(
     user_name: user_name,
     password: initial_password,
     password_reset_required: true
   )
   @logger.info("User '#{user_name}' created successfully.")
   response.user.user_id
 rescue Aws::IAM::Errors::EntityAlreadyExists
   @logger.error("Error creating user '#{user_name}': user already exists.")
   nil
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error creating user '#{user_name}': #{e.message}")
```

```
    nil
  end
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn create_user(client: &iamClient, user_name: &str) -> Result<User,
 iamError> {
    let response = client.create_user().user_name(user_name).send().await?;

    Ok(response.user.unwrap())
}
```

- Pour plus de détails sur l'API, voir CreateUserla section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d’être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
public func createUser(name: String) async throws -> String {
    let input = CreateUserInput(
        userName: name
    )
    do {
        let output = try await client.createUser(input: input)
        guard let user = output.user else {
            throw ServiceHandlerError.noSuchUser
        }
        guard let id = user.userId else {
            throw ServiceHandlerError.noSuchUser
        }
        return id
    } catch {
        throw error
    }
}
```

- Pour plus de détails sur l'API, reportez-vous CreateUserà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`CreateVirtualMfaDevice`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`CreateVirtualMfaDevice`.

CLI

AWS CLI

Pour créer un périphérique MFA virtuel

Cet exemple crée un nouveau périphérique MFA virtuel appelé. `BobsMFADevice` Il crée un fichier qui contient les informations de bootstrap appelées `QRCode.png` et le place dans le `C:/` répertoire. La méthode bootstrap utilisée dans cet exemple est`QRCodePNG`.

```
aws iam create-virtual-mfa-device \
    --virtual-mfa-device-name BobsMFADevice \
    --outfile C:/QRCode.png \
    --bootstrap-method QRCodePNG
```

Sortie :

```
{
    "VirtualMFADevice": {
        "SerialNumber": "arn:aws:iam::210987654321:mfa/BobsMFADevice"
}
```

Pour plus d'informations, consultez Utilisation de l'authentification multifacteur (MFA) dans AWS dans le AWS Guide de l'utilisateur IAM.

- Pour plus de détails sur l'API, reportez-vous CreateVirtualMfaDeviceà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée un nouveau périphérique MFA virtuel. Les lignes 2 et 3 extraient la **`Base32StringSeed`** valeur dont le logiciel MFA virtuel a besoin pour créer un compte (comme alternative au code QR). Après avoir configuré le programme avec la valeur, obtenez deux codes d'authentification séquentiels auprès du programme. Enfin, utilisez la dernière commande pour relier le périphérique MFA virtuel à l'utilisateur IAM **`Bob`** et synchroniser le compte avec les deux codes d'authentification.

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$SR = New-Object System.IO.StreamReader($Device.Base32StringSeed)
$base32stringseed = $SR.ReadToEnd()
$base32stringseed
CZWZMCQNW4DEXAMPLE3VOUGXJFZYSUW7EXAMPLECR4NJFD65GX2SLUDW2EXAMPLE
```

Sortie :

```
-- Pause here to enter base-32 string seed code into virtual MFA program to
 register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

Exemple 2 : Cet exemple crée un nouveau périphérique MFA virtuel. Les lignes 2 et 3 extraient la **`QRCodePNG`** valeur et l'écrivaient dans un fichier. Cette image peut être numérisée par le logiciel MFA virtuel pour créer un compte (au lieu de saisir manuellement la valeur Base32StringSeed). Après avoir créé le compte dans votre programme MFA virtuel, obtenez deux codes d'authentification séquentiels et saisissez-les dans les dernières commandes pour relier le dispositif MFA virtuel à l'utilisateur IAM et synchroniser le compte. **`Bob`**

```
$Device = New-IAMVirtualMFADevice -VirtualMFADeviceName BobsMFADevice
$BR = New-Object System.IO.BinaryReader($Device.QRCodePNG)
$BR.ReadBytes($BR.BaseStream.Length) | Set-Content -Encoding Byte -Path
 QRCode.png
```

Sortie :

```
 -- Pause here to scan PNG with virtual MFA program to register account. --

Enable-IAMMFADevice -SerialNumber $Device.SerialNumber -UserName Bob -
AuthenticationCode1 123456 -AuthenticationCode2 789012
```

- Pour plus de détails sur l'API, reportez-vous CreateVirtualMfaDeviceà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeactivateMfaDevice`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeactivateMfaDevice`.

CLI

AWS CLI

Pour désactiver un appareil MFA

Cette commande désactive le périphérique MFA virtuel avec l'`arn:aws:iam::210987654321:mfa/BobsMFADevice`ARN associé à l'utilisateur. `Bob`

```
aws iam deactivate-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez Utilisation de l'authentification multifacteur (MFA) dans AWS dans le AWS Guide de l'utilisateur IAM.

- Pour plus de détails sur l'API, consultez DeactivateMfaDevice in AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cette commande désactive le périphérique MFA matériel associé à l'**Bob**utilisateur qui possède le numéro de série. **123456789012**

```
Disable-IAMMFADevice -UserName "Bob" -SerialNumber "123456789012"
```

Exemple 2 : Cette commande désactive le périphérique MFA virtuel associé à l'**David**utilisateur qui possède l'ARN. **arn:aws:iam::210987654321:mfa/David** Notez que le périphérique MFA virtuel n'est pas supprimé du compte. Le périphérique virtuel est toujours présent et apparaît dans la sortie de la **Get-IAMVirtualMFADevice** commande. Avant de créer un nouveau périphérique MFA virtuel pour le même utilisateur, vous devez supprimer l'ancien à l'aide de la **Remove-IAMVirtualMFADevice** commande.

```
Disable-IAMMFADevice -UserName "David" -SerialNumber
 "arn:aws:iam::210987654321:mfa/David"
```

- Pour plus de détails sur l'API, voir DeactivateMfaDevice in AWS Tools for PowerShell Cmdlet Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteAccessKey`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteAccessKey`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Créer un groupe et ajouter un utilisateur
- Créer un utilisateur et assumer d'un rôle
- Créer des utilisateurs en lecture seule et en lecture-écriture
- Gérer des clés d'accès

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
```

```
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
 DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, voir DeleteAccessKey in AWS SDK for .NET API Reference.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
```

```
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an WS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  if [[ -z "$access_key" ]]; then
```

```
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}
```

- Pour plus de détails sur l'API, voir DeleteAccessKey in AWS CLI Command Reference.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::deleteAccessKey(const Aws::String &userName,
```

```
                                    const Aws::String &accessKeyID,
                                    const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);

    auto outcome = iam.DeleteAccessKey(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting access key " << accessKeyID << " from user "
                  << userName << ": " << outcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully deleted access key " << accessKeyID
                  << " for IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, voir DeleteAccessKey in AWS SDK for C++ API Reference.

CLI

AWS CLI

Pour supprimer une clé d’accès d’un utilisateur IAM

La commande `delete-access-key` suivante supprime la clé d’accès spécifiée (un ID de clé d’accès rapide et une clé d’accès secrète) de l’utilisateur IAM nommé Bob.

```
aws iam delete-access-key \
    --access-key-id AKIDPMS9RO4H3FEXAMPLE \
    --user-name Bob
```

Cette commande ne produit aucun résultat.

Pour répertorier les clés d'accès définies pour un utilisateur IAM, utilisez la commande `list-access-keys`.

Pour de plus amples informations, veuillez consulter Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir DeleteAccessKey in AWS CLI Command Reference.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(userName string, keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
 &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}
```

- Pour plus de détails sur l'API, voir DeleteAccessKey in AWS SDK for Go API Reference.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteAccessKeyRequest;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAccessKey {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username> <accessKey>\s

                Where:
                    username - The name of the user.\s
                    accessKey - The access key ID for the secret access key you
 want to delete.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
```

```
        }

        String username = args[0];
        String accessKey = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        deleteKey(iam, username, accessKey);
        iam.close();
    }

    public static void deleteKey(IamClient iam, String username, String
 accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, voir DeleteAccessKey in AWS SDK for Java 2.x API Reference.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Supprimez la clé d'accès.

```
import { DeleteAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const deleteAccessKey = (userName, accessKeyId) => {
  const command = new DeleteAccessKeyCommand({
    AccessKeyId: accessKeyId,
    UserName: userName,
  });

  return client.send(command);
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, voir DeleteAccessKey in AWS SDK for JavaScript API Reference.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
  UserName: "USER_NAME",
};

iam.deleteAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, voir DeleteAccessKey in AWS SDK for JavaScript API Reference.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun deleteKey(
    userNameVal: String,
    accessKey: String
) {
    val request =
        DeleteAccessKeyRequest {
            accessKeyId = accessKey
            userName = userNameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccessKey(request)
        println("Successfully deleted access key $accessKey from $userNameVal")
    }
}
```

- Pour plus de détails sur l'API, voir DeleteAccessKey in AWS SDK for Kotlin API reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime la paire de clés AWS d'accès avec l'ID **AKIAIOSFODNN7EXAMPLE** de clé de l'utilisateur nommé**Bob**.

```
Remove-IAMAccessKey -AccessKeyId AKIAIOSFODNN7EXAMPLE -UserName Bob -Force
```

- Pour plus de détails sur l'API, consultez DeleteAccessla section Key in AWS Tools for PowerShell Cmdlet Reference.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

- Pour plus de détails sur l'API, consultez le manuel de référence de l'API DeleteAccessKey in AWS SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, désactive et supprime les clés d'accès.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "AccessKeyManager"
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity => e
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
 #{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded => e
    @logger.error("Error creating access key: limit exceeded. Cannot create
 more.")
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
```

```
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: "Inactive"
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, voir DeleteAccessKey in AWS SDK for Ruby API Reference.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn delete_access_key(
    client: &iamClient,
    user: &User,
    key: &AccessKey,
) -> Result<(), iamError> {
    loop {
        match client
            .delete_access_key()
            .user_name(user.user_name())
            .access_key_id(key.access_key_id())
            .send()
            .await
        {
            Ok(_) => {
                break;
            }
            Err(e) => {
                println!("Can't delete the access key: {:?}", e);
                sleep(Duration::from_secs(2)).await;
            }
        }
    }
    Ok(())
}
```

- Pour plus de détails sur l'API, voir DeleteAccessKey in AWS SDK for Rust API reference.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
public func deleteAccessKey(user: IAMClientTypes.User? = nil,
                            key: IAMClientTypes.AccessKey) async throws {
    let userName: String?

    if user != nil {
        userName = user!.userName
    } else {
        userName = nil
    }

    let input = DeleteAccessKeyInput(
        accessKeyId: key.accessKeyId,
        userName: userName
    )
    do {
        _ = try await iamClient.deleteAccessKey(input: input)
    } catch {
        throw error
    }
}
```

- Pour plus de détails sur l'API, voir DeleteAccessKey in AWS SDK for Swift API reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteAccountAlias`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteAccountAlias`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer votre compte

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::deleteAccountAlias(const Aws::String &accountAlias,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccountAliasRequest request;
    request.SetAccountAlias(accountAlias);

    const auto outcome = iam.DeleteAccountAlias(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting account alias " << accountAlias << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted account alias " << accountAlias <<
                  std::endl;
    }
```

```
    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, voir DeleteAccountAlias dans le guide de référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour supprimer un alias de compte

La commande `delete-account-alias` suivante supprime les alias `mycompany` du compte actuel.

```
aws iam delete-account-alias \
    --account-alias mycompany
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Votre identifiant de AWS compte et son alias dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir DeleteAccountAlias dans AWS CLI la référence des commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.DeleteAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
```

```
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAccountAlias {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <alias>\s

                Where:
                    alias - The account alias to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMAccountAlias(iam, alias);
        iam.close();
    }

    public static void deleteIAMAccountAlias(IamClient iam, String alias) {
        try {
            DeleteAccountAliasRequest request =
 DeleteAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();
```

```
            iam.deleteAccountAlias(request);
            System.out.println("Successfully deleted account alias " + alias);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Pour plus de détails sur l'API, voir DeleteAccountAlias dans le guide de référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Supprimez l'alias de compte.

```
import { DeleteAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias
 */
export const deleteAccountAlias = (alias) => {
  const command = new DeleteAccountAliasCommand({ AccountAlias: alias });

  return client.send(command);
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, voir DeleteAccountAlias dans le guide de référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, voir DeleteAccountAlias dans le guide de référence des AWS SDK for JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun deleteIAMAccountAlias(alias: String) {
    val request =
        DeleteAccountAliasRequest {
            accountAlias = alias
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccountAlias(request)
        println("Successfully deleted account alias $alias")
    }
}
```

- Pour plus de détails sur l'API, voir DeleteAccountAlias dans le AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime l'alias de compte de votre Compte AWS. La page de connexion de l'utilisateur avec l'alias https://mycompanyaws.signin.aws.amazon.com/console ne fonctionne plus. Vous devez plutôt utiliser l'URL d'origine avec votre numéro d' Compte AWS identification sur https ://.signin.aws.amazon.com/console. <accountidnumber>

```
Remove-IAMAccountAlias -AccountAlias mycompanyaws
```

- Pour plus de détails sur l'API, voir DeleteAccountAlias dans la référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise
```

- Pour plus de détails sur l'API, voir DeleteAccountAlias in AWS SDK for Python (Boto3) API Reference.

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez, créez et supprimez des alias de compte.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info("Account aliases are:")
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info("No account aliases found.")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, voir DeleteAccountAlias dans le guide de référence des AWS SDK for Ruby API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteAccountPasswordPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteAccountPasswordPolicy`.

CLI

AWS CLI

Pour supprimer la politique de mot de passe du compte actuel

La `delete-account-password-policy` commande suivante supprime la politique de mot de passe du compte actuel.

```
aws iam delete-account-password-policy
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Définition d'une politique de mot de passe du compte pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir DeleteAccountPasswordPolicyla section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime la politique de mot de passe pour le Compte AWS et rétablit toutes les valeurs par défaut d'origine. Si aucune politique de mot de passe n'existe actuellement, le message d'erreur suivant s'affiche : PasswordPolicy Impossible de trouver la politique de compte portant le nom.

```
Remove-IAMAccountPasswordPolicy
```

- Pour plus de détails sur l'API, consultez la section DeleteAccountPasswordPolicyRéférence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteGroup`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteGroup`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Créer un groupe et ajouter un utilisateur

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
```

```
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteGroupà la section Référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour supprimer un groupe IAM

La commande `delete-group` suivante supprime un groupe IAM nommé `MyTestGroup`.

```
aws iam delete-group \
    --group-name MyTestGroup
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Suppression d'un groupe d'utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteGroupà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { DeleteGroupCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} groupName
 */
export const deleteGroup = async (groupName) => {
  const command = new DeleteGroupCommand({
    GroupName: groupName,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Pour plus de détails sur l'API, reportez-vous DeleteGroupà la section Référence des AWS SDK for JavaScript API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime le groupe IAM nommé. **`MyTestGroup`** La première commande supprime tous les utilisateurs IAM membres du groupe, et la seconde supprime le groupe IAM. Les deux commandes fonctionnent sans aucune demande de confirmation.

```
(Get-IAMGroup -GroupName MyTestGroup).Users | Remove-IAMUserFromGroup -GroupName
 MyTestGroup -Force
Remove-IAMGroup -GroupName MyTestGroup -Force
```

- Pour plus de détails sur l'API, reportez-vous DeleteGroupà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteGroupPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteGroupPolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Créer un groupe et ajouter un utilisateur

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
```

```
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, consultez DeleteGroupla section Politique dans le Guide de référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour supprimer une politique d'un groupe IAM

La commande `delete-group-policy` suivante supprime la politique nommée `ExamplePolicy` du groupe nommé `Admins`.

```
aws iam delete-group-policy \
    --group-name Admins \
    --policy-name ExamplePolicy
```

Cette commande ne produit aucun résultat.

Pour voir les politiques attachées à un groupe, utilisez la commande `list-group-policies`.

Pour plus d'informations, consultez Gestion des politiques IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez DeleteGroupla section Politique dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime la politique en ligne nommée dans le **`TesterPolicy`** groupe IAM. **`Testers`** Les utilisateurs de ce groupe perdent immédiatement les autorisations définies dans cette politique.

```
Remove-IAMGroupPolicy -GroupName Testers -PolicyName TestPolicy
```

- Pour plus de détails sur l'API, consultez DeleteGroupla section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteInstanceProfile`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteInstanceProfile`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Créer et gérer un service résilient

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
```

```
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
    /// <param name="profileName">The name of the profile to delete.</param>
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
    {
        try
        {
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
```

```
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous à la section DeleteInstanceProfil dans le AWS SDK for .NET manuel de référence des API.

CLI

AWS CLI

Pour supprimer un profil d'instance

La commande `delete-instance-profile` suivante supprime un profil d'instance nommé `ExampleInstanceProfile`.

```
aws iam delete-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

Cette commande ne produit aucun résultat.

Pour de plus amples informations, consultez Utilisation de profils d'instance dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section DeleteInstanceProfil dans AWS CLI la référence des commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
  const client = new IAMClient({});
  await client.send(
    new DeleteInstanceProfileCommand({
      InstanceProfileName: NAMES.instanceProfileName,
    }),
  );
```

- Pour plus de détails sur l'API, reportez-vous à la section DeleteInstanceProfil dans le AWS SDK for JavaScript manuel de référence des API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime le profil d'instance EC2 nommé. **`MyAppInstanceProfile`** La première commande détache tous les rôles du profil d'instance, puis la deuxième commande supprime le profil d'instance.

```
(Get-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile).Roles |
 Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyAppInstanceProfile
Remove-IAMInstanceProfile -InstanceProfileName MyAppInstanceProfile
```

- Pour plus de détails sur l'API, consultez la section DeleteInstanceProfil dans la référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple supprime le rôle du profil d'instance, détache toutes les politiques associées au rôle et supprime toutes les ressources.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"
```

```
   def delete_instance_profile(self, profile_name, role_name):
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete instance profile {profile_name} or detach "
                    f"policies and delete role {role_name}: {err}"
                )
```

- Pour plus de détails sur l'API, voir DeleteInstanceProfile in AWS SDK for Python (Boto3) API Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteLoginProfile`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteLoginProfile`.

CLI

AWS CLI

Pour supprimer le mot de passe d'un utilisateur IAM

La `delete-login-profile` commande suivante supprime le mot de passe de l'utilisateur IAM nommé. `Bob`

```
aws iam delete-login-profile \
    --user-name Bob
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Gestion des mots de passe pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section DeleteLoginProfil dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime le profil de connexion de l'utilisateur IAM nommé. **`Bob`** Cela empêche l'utilisateur de se connecter à la AWS console. Cela n'empêche pas l'utilisateur d'exécuter des appels de AWS CLI ou d'API AWS à l'aide de clés d'accès qui peuvent toujours être associées au compte utilisateur. PowerShell

```
Remove-IAMLoginProfile -UserName Bob
```

- Pour plus de détails sur l'API, consultez la section DeleteLoginProfil dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteOpenIdConnectProvider`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteOpenIdConnectProvider`.

CLI

AWS CLI

Pour supprimer un fournisseur d'identité IAM OpenID Connect

Cet exemple supprime le fournisseur IAM OIDC qui se connecte au fournisseur. `example.oidcprovider.com`

```
aws iam delete-open-id-connect-provider \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Création de fournisseurs d'identité OpenID Connect (OIDC) dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir DeleteOpenIdConnectProvider dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime le fournisseur IAM OIDC qui se connecte au fournisseur. **`example.oidcprovider.com`** Assurez-vous de mettre à jour ou de supprimer tous les rôles

qui font référence à ce fournisseur dans l'**`Principal`**élément de la politique de confiance du rôle.

```
Remove-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com
```

- Pour plus de détails sur l'API, consultez la section DeleteOpenIdConnectFournisseur dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeletePolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeletePolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Créer un utilisateur et assumer d'un rôle
- Créer des utilisateurs en lecture seule et en lecture-écriture
- Gérer les politiques

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
```

```
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS SDK for .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
```

```
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an WS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1
```

```
  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::deletePolicy(const Aws::String &policyArn,
                               const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::DeletePolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.DeletePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting policy with arn " << policyArn << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted policy with arn " << policyArn
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour supprimer une politique IAM

Cet exemple supprime la politique dont l'ARN est `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam delete-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Cette commande ne produit aucun résultat.

Pour de plus amples informations, veuillez consulter Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(policyArn string) error {
 _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS SDK for Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.DeletePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeletePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <policyARN>\s

                Where:
                    policyARN - A policy ARN value to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String policyARN = args[0];
        Region region = Region.AWS_GLOBAL;
```

```
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMPolicy(iam, policyARN);
        iam.close();
    }

    public static void deleteIAMPolicy(IamClient iam, String policyARN) {
        try {
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(policyARN)
                    .build();

            iam.deletePolicy(request);
            System.out.println("Successfully deleted the policy");

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Supprimez la politique.

```
import { DeletePolicyCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const deletePolicy = (policyArn) => {
  const command = new DeletePolicyCommand({ PolicyArn: policyArn });
  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS SDK for JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun deleteIAMPolicy(policyARNVal: String?) {
    val request =
        DeletePolicyRequest {
            policyArn = policyARNVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deletePolicy(request)
        println("Successfully deleted $policyARNVal")
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime la politique dont l'ARN est **`arn:aws:iam::123456789012:policy/MySamplePolicy`** l'ARN. Avant de pouvoir supprimer la politique, vous devez d'abord supprimer toutes les versions, à l'exception de la version par défaut, en exécutant la commande**`Remove-IAMPolicyVersion`**. Vous devez également dissocier la politique de tous les utilisateurs, groupes ou rôles IAM.

```
Remove-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Exemple 2 : cet exemple supprime une politique en supprimant d'abord toutes les versions de stratégie autres que celles par défaut, en la détachant de toutes les entités IAM attachées, puis en supprimant la politique elle-même. La première ligne permet de récupérer l'objet de politique. La deuxième ligne récupère toutes les versions de politique qui ne sont pas signalées comme version par défaut dans une collection, puis supprime chaque politique de la collection. La troisième ligne récupère tous les utilisateurs, groupes et rôles IAM auxquels la politique est attachée. Les lignes quatre à six détachent la politique de chaque entité attachée. La dernière ligne utilise cette commande pour supprimer la politique gérée ainsi que la version par défaut restante. L'exemple inclut le paramètre **`-Force`** switch sur toute ligne qui en a besoin pour supprimer les demandes de confirmation.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
$attached = Get-IAMEntitiesForPolicy -PolicyArn $pol.Arn
$attached.PolicyGroups | Unregister-IAMGroupPolicy -PolicyArn $pol.arn
$attached.PolicyRoles | Unregister-IAMRolePolicy -PolicyArn $pol.arn
$attached.PolicyUsers | Unregister-IAMUserPolicy -PolicyArn $pol.arn
Remove-IAMPolicy $pol.Arn -Force
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

- Pour plus de détails sur l'API, consultez DeletePolicyle AWS manuel de référence de l'API SDK for Python (Boto3).

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn delete_policy(client: &iamClient, policy: Policy) -> Result<(),
 iamError> {
    client
        .delete_policy()
        .policy_arn(policy.arn.unwrap())
        .send()
        .await?;
    Ok(())
}
```

- Pour plus de détails sur l'API, voir DeletePolicyla section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    public func deletePolicy(policy: IAMClientTypes.Policy) async throws {
        let input = DeletePolicyInput(
            policyArn: policy.arn
        )
        do {
            _ = try await iamClient.deletePolicy(input: input)
        } catch {
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous DeletePolicyà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeletePolicyVersion`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeletePolicyVersion`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Gérer les politiques
- Restaurer une version de stratégie

CLI

AWS CLI

Pour supprimer une version d'une politique gérée

Cet exemple supprime la version identifiée comme étant dans `v2` la politique dont l'ARN est `arn:aws:iam::123456789012:policy/MySamplePolicy` l'ARN.

```
aws iam delete-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Cette commande ne produit aucun résultat.

Pour de plus amples informations, veuillez consulter Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir DeletePolicyVersion dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime la version identifiée comme dans la politique dont **v2** l'ARN est **arn:aws:iam::123456789012:policy/MySamplePolicy** l'ARN.

```
Remove-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MySamplePolicy -VersionID v2
```

Exemple 2 : Cet exemple supprime une politique en supprimant d'abord toutes les versions de stratégie autres que celles par défaut, puis en supprimant la politique elle-même. La première ligne permet de récupérer l'objet de politique. La deuxième ligne récupère toutes les versions de politique qui ne sont pas signalées comme étant par défaut dans une collection, puis utilise cette commande pour supprimer chaque politique de la collection. La dernière ligne supprime la politique elle-même ainsi que la version par défaut restante. Notez que pour supprimer correctement une politique gérée, vous devez également la détacher de tous les utilisateurs, groupes ou rôles à l'aide des **Unregister-IAMRolePolicy** commandes **Unregister-IAMUserPolicyUnregister-IAMGroupPolicy**, et. Consultez l'exemple de l'**Remove-IAMPolicy**applet de commande.

```
$pol = Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
Get-IAMPolicyVersions -PolicyArn $pol.Arn | where {-not $_.IsDefaultVersion} |
 Remove-IAMPolicyVersion -PolicyArn $pol.Arn -force
Remove-IAMPolicy -PolicyArn $pol.Arn -force
```

- Pour plus de détails sur l'API, consultez DeletePolicyla section Version dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **DeleteRole** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteRole`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Créer un utilisateur et assumer d'un rôle
- Gérer les rôles

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
{ RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des AWS SDK for .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
```

```
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an WS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")
```

```
  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des AWS CLI commandes.

CLI

AWS CLI

Pour supprimer un rôle IAM

La commande `delete-role` suivante supprime le rôle nommé `Test-Role`.

```
aws iam delete-role \
    --role-name Test-Role
```

Cette commande ne produit aucun résultat.

Pour pouvoir supprimer un rôle, vous devez le supprimer de tout profil d'instance (`remove-role-from-instance-profile`), détacher toutes les politiques gérées (`detach-role-policy`) et supprimer toutes les politiques intégrées attachées au rôle (`delete-role-policy`).

Pour plus d'informations, consultez Création de rôles IAM et Utilisation de profils d'instance dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteRole(context.TODO(), &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des AWS SDK for Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Supprimez le rôle.

```
import { DeleteRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const deleteRole = (roleName) => {
  const command = new DeleteRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des AWS SDK for JavaScript API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime le rôle nommé dans le **MyNewRole** compte IAM actuel. Avant de pouvoir supprimer le rôle, vous devez d'abord utiliser la **Unregister-IAMRolePolicy** commande pour détacher les politiques gérées. Les politiques intégrées sont supprimées avec le rôle.

```
Remove-IAMRole -RoleName MyNewRole
```

Exemple 2 : Cet exemple détache toutes les politiques gérées du rôle nommé, **MyNewRole** puis supprime le rôle. La première ligne récupère toutes les politiques gérées associées au rôle en tant que collection, puis détache chaque politique de la collection du rôle. La deuxième ligne supprime le rôle lui-même. Les politiques intégrées sont supprimées en même temps que le rôle.

```
Get-IAMAttachedRolePolicyList -RoleName MyNewRole | Unregister-IAMRolePolicy -
RoleName MyNewRole
Remove-IAMRole -RoleName MyNewRole
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def delete_role(role_name):
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

- Pour plus de détails sur l'API, consultez DeleteRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
  # Deletes a role and its attached policies.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    begin
      # Detach and delete attached policies
      @iam_client.list_attached_role_policies(role_name: role_name).each do |
response|
        response.attached_policies.each do |policy|
          @iam_client.detach_role_policy({
                                           role_name: role_name,
                                           policy_arn: policy.policy_arn
                                         })
          # Check if the policy is a customer managed policy (not AWS managed)
          unless policy.policy_arn.include?("aws:policy/")
            @iam_client.delete_policy({ policy_arn: policy.policy_arn })
            @logger.info("Deleted customer managed policy
 #{policy.policy_name}.")
          end
        end
      end

      # Delete the role
      @iam_client.delete_role({ role_name: role_name })
      @logger.info("Deleted role #{role_name}.")
    rescue Aws::IAM::Errors::ServiceError => e
      @logger.error("Couldn't detach policies and delete role #{role_name}.
 Here's why:")
```

```
      @logger.error("\t#{e.code}: #{e.message}")
      raise
    end
  end
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn delete_role(client: &iamClient, role: &Role) -> Result<(), iamError>
 {
    let role = role.clone();
    while client
        .delete_role()
        .role_name(role.role_name())
        .send()
        .await
        .is_err()
    {
        sleep(Duration::from_secs(2)).await;
    }
    Ok(())
}
```

- Pour plus de détails sur l'API, voir DeleteRolela section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
public func deleteRole(role: IAMClientTypes.Role) async throws {
    let input = DeleteRoleInput(
        roleName: role.roleName
    )
    do {
        _ = try await iamClient.deleteRole(input: input)
    } catch {
        throw error
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteRoleà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteRolePermissionsBoundary`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteRolePermissionsBoundary`.

CLI

AWS CLI

Pour supprimer une limite d'autorisations d'un rôle IAM

L'`delete-role-permissions-boundary`exemple suivant supprime la limite des autorisations pour le rôle IAM spécifié. Pour appliquer une limite d'autorisations à un rôle, utilisez la `put-role-permissions-boundary` commande.

```
aws iam delete-role-permissions-boundary \
    --role-name lambda-application-role
```

Cette commande ne produit aucun résultat.

Pour de plus amples informations, veuillez consulter Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir DeleteRolePermissionsBoundaryla section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple montre comment supprimer la limite d'autorisation attachée à un rôle IAM.

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- Pour plus de détails sur l'API, consultez la section DeleteRolePermissionsBoundaryRéférence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteRolePolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteRolePolicy`.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, consultez DeleteRolela section Politique dans le Guide de référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour supprimer une politique d'un rôle IAM

La commande `delete-role-policy` suivante supprime la politique nommée `ExamplePolicy` du rôle nommé `Test-Role`.

```
aws iam delete-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez Modification d'un rôle dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez DeleteRolela section Politique dans AWS CLI Command Reference.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { DeleteRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 */
export const deleteRolePolicy = (roleName, policyName) => {
  const command = new DeleteRolePolicyCommand({
    RoleName: roleName,
    PolicyName: policyName,
  });
  return client.send(command);
};
```

```
```

- Pour plus de détails sur l'API, consultez DeleteRolela section Politique dans le Guide de référence des AWS SDK for JavaScript API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime la politique **`S3AccessPolicy`** intégrée intégrée au rôle IAM. **`S3BackupRole`**

```
Remove-IAMRolePolicy -PolicyName S3AccessPolicy -RoleName S3BackupRole
```

- Pour plus de détails sur l'API, consultez DeleteRolela section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteSAMLProvider`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteSAMLProvider`.

CLI

AWS CLI

Pour supprimer un fournisseur SAML

Cet exemple supprime le fournisseur IAM SAML 2.0 dont l'ARN est `arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`.

```
aws iam delete-saml-provider \
--saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez Création de fournisseurs d'identité SAML IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus d'informations sur l'API, consultez DeleteSAMLProvider dans la Référence des commandes AWS CLI .

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { DeleteSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} providerArn
 * @returns
 */
export const deleteSAMLProvider = async (providerArn) => {
  const command = new DeleteSAMLProviderCommand({
    SAMLProviderArn: providerArn,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Pour les détails de l'API, consultez DeleteSAMLProvider dans la Référence de l'API AWS SDK for JavaScript .

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime le fournisseur IAM SAML 2.0 dont l'ARN est. **`arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`**

```
Remove-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFSProvider
```

- Pour plus de détails sur l'API, consultez DeleteSamlProvider dans la référence des applets de commande.AWS Tools for PowerShell

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteServerCertificate`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteServerCertificate`.

C++

Kit de développement logiciel (SDK) for C++

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::deleteServerCertificate(const Aws::String &certificateName,
                                          const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::DeleteServerCertificateRequest request;
    request.SetServerCertificateName(certificateName);

    const auto outcome = iam.DeleteServerCertificate(request);
    bool result = true;
```

```
    if (!outcome.IsSuccess()) {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error deleting server certificate " << certificateName
 <<
                      ": " << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << certificateName
                      << "' not found." << std::endl;
        }
    }
    else {
        std::cout << "Successfully deleted server certificate " <<
 certificateName
                  << std::endl;
    }

    return result;
}
```

- Pour plus de détails sur l'API, consultez la section DeleteServerCertificat dans la référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour supprimer un certificat de serveur de votre AWS compte

La `delete-server-certificate` commande suivante supprime le certificat de serveur spécifié de votre AWS compte.

```
aws iam delete-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

Cette commande ne produit aucun résultat.

Pour répertorier les certificats de serveur disponibles dans votre AWS compte, utilisez la `list-server-certificates` commande.

Pour de plus amples informations, veuillez consulter Gestion des certificats de serveur dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section DeleteServerCertificat dans AWS CLI la référence des commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Supprimez un certificat de serveur.

```
import { DeleteServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 */
export const deleteServerCertificate = (certName) => {
  const command = new DeleteServerCertificateCommand({
    ServerCertificateName: certName,
  });

  return client.send(command);
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez la section DeleteServerCertificat dans la référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteServerCertificate(
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez la section DeleteServerCertificat dans la référence des AWS SDK for JavaScript API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime le certificat de serveur nommé**MyServerCert**.

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- Pour plus de détails sur l'API, consultez la section DeleteServerCertificat dans la référence des AWS Tools for PowerShell applets de commande.

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez, mettez à jour et supprimez les certificats de serveur.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "ServerCertificateManager"
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key,
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates
```

```
    if response.server_certificate_metadata_list.empty?
      @logger.info("No server certificates found.")
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
 #{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, consultez la section DeleteServerCertificat dans la référence des AWS SDK for Ruby API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteServiceLinkedRole`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteServiceLinkedRole`.

CLI

AWS CLI

Pour supprimer un rôle lié à un service

L'exemple `delete-service-linked-role` suivant supprime le rôle lié à un service spécifié dont vous n'avez plus besoin. La suppression s'effectue de manière asynchrone. Vous pouvez vérifier le statut de la suppression et si elle est terminée à l'aide de la commande `get-service-linked-role-deletion-status`.

```
aws iam delete-service-linked-role \
    --role-name AWSServiceRoleForLexBots
```

Sortie :

```
{
    "DeletionTaskId": "task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE"
}
```

Pour plus d'informations, consultez Utilisation des rôles liés à un service dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteServiceLinkedRoleà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteServiceLinkedRoleà la section Référence des AWS SDK for Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { DeleteServiceLinkedRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const deleteServiceLinkedRole = (roleName) => {
  const command = new DeleteServiceLinkedRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous DeleteServiceLinkedRoleà la section Référence des AWS SDK for JavaScript API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple a supprimé le rôle lié au service. Notez que si le service utilise toujours ce rôle, cette commande entraîne un échec.

```
Remove-IAMServiceLinkedRole -RoleName
 AWSServiceRoleForAutoScaling_RoleNameEndsWithThis
```

- Pour plus de détails sur l'API, reportez-vous DeleteServiceLinkedRoleà la section Référence des AWS Tools for PowerShell applets de commande.

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# Deletes a service-linked role.
#
# @param role_name [String] The name of the role to delete.
def delete_service_linked_role(role_name)
  response = @iam_client.delete_service_linked_role(role_name: role_name)
  task_id = response.deletion_task_id
  check_deletion_status(role_name, task_id)
rescue Aws::Errors::ServiceError => e
  handle_deletion_error(e, role_name)
end

private

# Checks the deletion status of a service-linked role
#
# @param role_name [String] The name of the role being deleted
# @param task_id [String] The task ID for the deletion process
def check_deletion_status(role_name, task_id)
  loop do
    response = @iam_client.get_service_linked_role_deletion_status(
      deletion_task_id: task_id)
    status = response.status
    @logger.info("Deletion of #{role_name} #{status}.")
    break if %w[SUCCEEDED FAILED].include?(status)
    sleep(3)
  end
end

# Handles deletion error
#
# @param e [Aws::Errors::ServiceError] The error encountered during deletion
# @param role_name [String] The name of the role attempted to delete
```

```
  def handle_deletion_error(e, role_name)
    unless e.code == "NoSuchEntity"
      @logger.error("Couldn't delete #{role_name}. Here's why:")
      @logger.error("\t#{e.code}: #{e.message}")
      raise
    end
  end
```

- Pour plus de détails sur l'API, reportez-vous DeleteServiceLinkedRoleà la section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn delete_service_linked_role(
    client: &iamClient,
    role_name: &str,
) -> Result<(), iamError> {
    client
        .delete_service_linked_role()
        .role_name(role_name)
        .send()
        .await?;

    Ok(())
}
```

- Pour plus de détails sur l'API, voir DeleteServiceLinkedRolela section de référence de l'API AWS SDK for Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteSigningCertificate`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteSigningCertificate`.

CLI

AWS CLI

Pour supprimer un certificat de signature pour un utilisateur IAM

La `delete-signing-certificate` commande suivante supprime le certificat de signature spécifié pour l'utilisateur IAM nommé. Bob

```
aws iam delete-signing-certificate \
    --user-name Bob \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE
```

Cette commande ne produit aucun résultat.

Pour obtenir l'ID d'un certificat de signature, utilisez la `list-signing-certificates` commande.

Pour plus d'informations, consultez la section Gérer les certificats de signature dans le guide de l'utilisateur Amazon EC2.

- Pour plus de détails sur l'API, consultez la section DeleteSigningCertificat dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime le certificat de signature portant l'ID **`Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU`** de l'utilisateur IAM nommé. **`Bob`**

```
Remove-IAMSigningCertificate -UserName Bob -CertificateId
 Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
```

- Pour plus de détails sur l'API, consultez la section DeleteSigningCertificat dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteUser`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteUser`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Créer un groupe et ajouter un utilisateur
- Créer un utilisateur et assumer d'un rôle
- Créer des utilisateurs en lecture seule et en lecture-écriture

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });
```

```
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS SDK for .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_delete_user
```

```
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
```

```
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteUserRequest request;
    request.SetUserName(userName);
    auto outcome = iam.DeleteUser(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting IAM user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;;
    }
    else {
        std::cout << "Successfully deleted IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour supprimer un utilisateur IAM

La commande `delete-user` suivante supprime l'utilisateur IAM nommé Bob du compte actuel.

```
aws iam delete-user \
    --user-name Bob
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Suppression d'un utilisateur IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(userName string) error {
 _, err := wrapper.IamClient.DeleteUser(context.TODO(), &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS SDK for Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteUserRequest;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <userName>\s

                Where:
                    userName - The name of the user to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
```

```
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMUser(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("Successfully deleted IAM user " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Supprimez l'utilisateur.

```
import { DeleteUserCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const deleteUser = (name) => {
  const command = new DeleteUserCommand({ UserName: name });
  return client.send(command);
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    console.log("User " + process.argv[2] + " does not exist.");
  } else {
```

```
    iam.deleteUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
    });
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS SDK for JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun deleteIAMUser(userNameVal: String) {
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }

    // To delete a user, ensure that the user's access keys are deleted first.
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteUser(request)
        println("Successfully deleted user $userNameVal")
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime l'utilisateur IAM nommé. **Bob**

```
Remove-IAMUser -UserName Bob
```

Exemple 2 : Cet exemple supprime l'utilisateur IAM nommé **Theresa** ainsi que tous les éléments qui doivent être supprimés en premier.

```
$name = "Theresa"

# find any groups and remove user from them
$groups = Get-IAMGroupForUser -UserName $name
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName $name -Force }

# find any inline policies and delete them
$inlinepols = Get-IAMUserPolicies -UserName $name
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 $name -Force}

# find any managed polices and detach them
$managedpols = Get-IAMAttachedUserPolicies -UserName $name
foreach ($pol in $managedpols) { Unregister-IAMUserPolicy -PolicyArn
 $pol.PolicyArn -UserName $name }

# find any signing certificates and delete them
$certs = Get-IAMSigningCertificate -UserName $name
foreach ($cert in $certs) { Remove-IAMSigningCertificate -CertificateId
 $cert.CertificateId -UserName $name -Force }

# find any access keys and delete them
$keys = Get-IAMAccessKey -UserName $name
foreach ($key in $keys) { Remove-IAMAccessKey -AccessKeyId $key.AccessKeyId -
UserName $name -Force }
```

```
# delete the user's login profile, if one exists - note: need to use try/catch to
 suppress not found error
try { $prof = Get-IAMLoginProfile -UserName $name -ea 0 } catch { out-null }
if ($prof) { Remove-IAMLoginProfile -UserName $name -Force }

# find any MFA device, detach it, and if virtual, delete it.
$mfa = Get-IAMMFADevice -UserName $name
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}

# finally, remove the user
Remove-IAMUser -UserName $name -Force
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
```

```
        raise
```

- Pour plus de détails sur l'API, consultez DeleteUserle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Deletes a user and their associated resources
 #
 # @param user_name [String] The name of the user to delete
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
   user.each do |key|
     @iam_client.delete_access_key({ access_key_id: key.access_key_id,
user_name: user_name })
     @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
   end

   @iam_client.delete_user(user_name: user_name)
   @logger.info("Deleted user '#{user_name}'.")
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error deleting user '#{user_name}': #{e.message}")
 end
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn delete_user(client: &iamClient, user: &User) -> Result<(),
 SdkError<DeleteUserError>> {
    let user = user.clone();
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;

    let response: Result<(), SdkError<DeleteUserError>> = loop {
        match client
            .delete_user()
            .user_name(user.user_name())
            .send()
            .await
        {
            Ok(_) => {
                break Ok(());
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    response
}
```

- Pour plus de détails sur l'API, voir DeleteUserla section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
public func deleteUser(user: IAMClientTypes.User) async throws {
    let input = DeleteUserInput(
        userName: user.userName
    )
    do {
        _ = try await iamClient.deleteUser(input: input)
    } catch {
        throw error
    }
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteUserPermissionsBoundary`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteUserPermissionsBoundary`.

CLI

AWS CLI

Pour supprimer une limite d'autorisation pour un utilisateur IAM

L'`delete-user-permissions-boundary`exemple suivant supprime la limite d'autorisations attachée à l'utilisateur IAM nommé. `intern` Pour appliquer une limite d'autorisation à un utilisateur, utilisez la `put-user-permissions-boundary` commande.

```
aws iam delete-user-permissions-boundary \
    --user-name intern
```

Cette commande ne produit aucun résultat.

Pour de plus amples informations, veuillez consulter Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteUserPermissionsBoundaryà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple montre comment supprimer la limite d'autorisation attachée à un utilisateur IAM.

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- Pour plus de détails sur l'API, reportez-vous DeleteUserPermissionsBoundaryà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteUserPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteUserPolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Créer un utilisateur et assumer d'un rôle

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, consultez DeleteUserla section Politique dans le Guide de référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour supprimer une politique d'un utilisateur IAM

La commande `delete-user-policy` suivante supprime la politique spécifiée de l'utilisateur IAM nommé Bob.

```
aws iam delete-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

Cette commande ne produit aucun résultat.

Pour obtenir une liste des politiques d'un utilisateur IAM, utilisez la commande `list-user-policies`.

Pour plus d'informations, consultez la section Création d'un utilisateur IAM dans votre AWS compte dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez DeleteUserla section Politique dans AWS CLI Command Reference.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}
```

```
// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(userName string, policyName string)
 error {
 _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
 &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

- Pour plus de détails sur l'API, consultez DeleteUserla section Politique dans le Guide de référence des AWS SDK for Go API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime la politique en ligne nommée **`AccessToEC2Policy`** qui est incorporée dans le nom de l'utilisateur IAM. **`Bob`**

```
Remove-IAMUserPolicy -PolicyName AccessToEC2Policy -UserName Bob
```

Exemple 2 : Cet exemple recherche toutes les politiques intégrées qui sont intégrées dans le nom de l'utilisateur IAM, **`Theresa`** puis les supprime.

```
$inlinepols = Get-IAMUserPolicies -UserName Theresa
foreach ($pol in $inlinepols) { Remove-IAMUserPolicy -PolicyName $pol -UserName
 Theresa -Force}
```

- Pour plus de détails sur l'API, consultez DeleteUserla section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Deletes a user and their associated resources
 #
 # @param user_name [String] The name of the user to delete
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
   user.each do |key|
     @iam_client.delete_access_key({ access_key_id: key.access_key_id,
user_name: user_name })
     @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
   end

   @iam_client.delete_user(user_name: user_name)
   @logger.info("Deleted user '#{user_name}'.")
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error deleting user '#{user_name}': #{e.message}")
 end
```

- Pour plus de détails sur l'API, consultez DeleteUserla section Politique dans le Guide de référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn delete_user_policy(
    client: &iamClient,
    user: &User,
    policy_name: &str,
) -> Result<(), SdkError<DeleteUserPolicyError>> {
    client
        .delete_user_policy()
        .user_name(user.user_name())
        .policy_name(policy_name)
        .send()
        .await?;

    Ok(())
}
```

- Pour plus de détails sur l'API, consultez DeleteUserla section Politique du AWS SDK pour la référence de l'API Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    func deleteUserPolicy(user: IAMClientTypes.User, policyName: String) async
throws {
        let input = DeleteUserPolicyInput(
            policyName: policyName,
            userName: user.userName
```

```
        )
        do {
            _ = try await iamClient.deleteUserPolicy(input: input)
        } catch {
            throw error
        }
    }
```

- Pour plus de détails sur l'API, consultez DeleteUserla section Politique du AWS SDK pour la référence à l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DeleteVirtualMfaDevice`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DeleteVirtualMfaDevice`.

CLI

AWS CLI

Pour supprimer un périphérique MFA virtuel

La `delete-virtual-mfa-device` commande suivante supprime le périphérique MFA spécifié du compte actuel.

```
aws iam delete-virtual-mfa-device \
    --serial-number arn:aws:iam::123456789012:mfa/MFATest
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Désactivation des appareils MFA dans AWS le guide de l'utilisateur IAM.

- Pour plus de détails sur l'API, reportez-vous DeleteVirtualMfaDeviceà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime le périphérique MFA virtuel IAM dont l'ARN est. **`arn:aws:iam::123456789012:mfa/bob`**

```
Remove-IAMVirtualMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/bob
```

Exemple 2 : Cet exemple vérifie si un dispositif MFA est attribué à l'utilisateur IAM Theresa. S'il en trouve un, l'appareil est désactivé pour l'utilisateur IAM. Si le périphérique est virtuel, il est également supprimé.

```
$mfa = Get-IAMMFADevice -UserName Theresa
if ($mfa) {
    Disable-IAMMFADevice -SerialNumber $mfa.SerialNumber -UserName $name
    if ($mfa.SerialNumber -like "arn:*") { Remove-IAMVirtualMFADevice -
SerialNumber $mfa.SerialNumber }
}
```

- Pour plus de détails sur l'API, reportez-vous DeleteVirtualMfaDeviceà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DetachGroupPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DetachGroupPolicy`.

CLI

AWS CLI

Pour détacher une politique d'un groupe

Cet exemple supprime la politique gérée avec l'ARN `arn:aws:iam::123456789012:policy/TesterAccessPolicy` du groupe appelé`Testers`.

```
aws iam detach-group-policy \
    --group-name Testers \
    --policy-arn arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Gestion des groupes IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez DetachGroupla section Politique dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple détache la politique de groupe géré dont l'ARN **`arn:aws:iam::123456789012:policy/TesterAccessPolicy`** provient du groupe nommé**`Testers`**.

```
Unregister-IAMGroupPolicy -GroupName Testers -PolicyArn
 arn:aws:iam::123456789012:policy/TesterAccessPolicy
```

Exemple 2 : Cet exemple trouve toutes les politiques gérées associées au groupe nommé **`Testers`** et les détache du groupe.

```
Get-IAMAttachedGroupPolicies -GroupName Testers | Unregister-IAMGroupPolicy -
Groupname Testers
```

- Pour plus de détails sur l'API, consultez DetachGroupla section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DetachRolePolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DetachRolePolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Créer un utilisateur et assumer d'un rôle
- Gérer les rôles

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /// <summary>
   /// Detach an IAM policy from an IAM role.
   /// </summary>
   /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
   /// <param name="roleName">The name of the IAM role.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
   {
       var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
       {
           PolicyArn = policyArn,
           RoleName = roleName,
       });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }
```

- Pour plus de détails sur l'API, consultez DetachRolela section Politique dans le Guide de référence des AWS SDK for .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
```

```
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
 }

 # Retrieve the calling parameters.
 while getopts "n:p:h" option; do
   case "${option}" in
     n) role_name="${OPTARG}" ;;
     p) policy_arn="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$role_name" ]]; then
   errecho "ERROR: You must provide a role name with the -n parameter."
   usage
   return 1
 fi

 if [[ -z "$policy_arn" ]]; then
   errecho "ERROR: You must provide a policy ARN with the -p parameter."
   usage
   return 1
 fi

 response=$(aws iam detach-role-policy \
   --role-name "$role_name" \
   --policy-arn "$policy_arn")

 local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, consultez DetachRolela section Politique dans AWS CLI Command Reference.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DetachRolePolicyRequest detachRequest;
    detachRequest.SetRoleName(roleName);
    detachRequest.SetPolicyArn(policyArn);

    auto detachOutcome = iam.DetachRolePolicy(detachRequest);
    if (!detachOutcome.IsSuccess()) {
        std::cerr << "Failed to detach policy " << policyArn << " from role "
                  << roleName << ": " << detachOutcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully detached policy " << policyArn << " from role
 "
                  << roleName << std::endl;
```

```
    }

    return detachOutcome.IsSuccess();
```

- Pour plus de détails sur l'API, consultez DetachRolela section Politique dans le Guide de référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour détacher une politique d’un rôle

Cet exemple supprime la politique gérée avec l’ARN `arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy` du rôle nommé `FedTesterRole`.

```
aws iam detach-role-policy \
    --role-name FedTesterRole \
    --policy-arn arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Cette commande ne produit aucun résultat.

Pour plus d’informations, consultez Modification d’un rôle dans le Guide de l’utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez DetachRolela section Politique dans AWS CLI Command Reference.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(roleName string, policyArn string)
 error {
 _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
 &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}
```

- Pour plus de détails sur l'API, consultez DetachRolela section Politique dans le Guide de référence des AWS SDK for Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.DetachRolePolicyRequest;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DetachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleName = args[0];
        String policyArn = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        detachPolicy(iam, roleName, policyArn);
        System.out.println("Done");
        iam.close();
    }
```

```
    public static void detachPolicy(IamClient iam, String roleName, String
 policyArn) {
        try {
            DetachRolePolicyRequest request = DetachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.detachRolePolicy(request);
            System.out.println("Successfully detached policy " + policyArn +
                    " from role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, consultez DetachRolela section Politique dans le Guide de référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Détachez la politique.

```
import { DetachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
```

```
 * @param {string} policyArn
 * @param {string} roleName
 */
export const detachRolePolicy = (policyArn, roleName) => {
  const command = new DetachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez DetachRolela section Politique dans le Guide de référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
```

```
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        var params = {
          PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
          RoleName: process.argv[2],
        };
        iam.detachRolePolicy(params, function (err, data) {
          if (err) {
            console.log("Unable to detach policy from role", err);
          } else {
            console.log("Policy detached from role successfully");
            process.exit();
          }
        });
      }
    });
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez DetachRolela section Politique dans le Guide de référence des AWS SDK for JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun detachPolicy(
    roleNameVal: String,
    policyArnVal: String
) {
    val request =
```

```
        DetachRolePolicyRequest {
            roleName = roleNameVal
            policyArn = policyArnVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.detachRolePolicy(request)
        println("Successfully detached policy $policyArnVal from role
 $roleNameVal")
    }
}
```

- Pour plus de détails sur l'API, consultez DetachRolela section Politique du AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple détache la politique de groupe géré dont l'ARN **`arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy`** provient du rôle nommé**`FedTesterRole`**.

```
Unregister-IAMRolePolicy -RoleName FedTesterRole -PolicyArn
 arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Exemple 2 : Cet exemple recherche toutes les politiques gérées associées au rôle nommé **`FedTesterRole`** et les détache du rôle.

```
Get-IAMAttachedRolePolicyList -RoleName FedTesterRole | Unregister-IAMRolePolicy
 -Rolename FedTesterRole
```

- Pour plus de détails sur l'API, consultez DetachRolela section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Détachez une politique d'un rôle à l'aide de l'objet de politique Boto3.

```
def detach_from_role(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).detach_role(RoleName=role_name)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

Détachez une politique d'un rôle à l'aide de l'objet de rôle Boto3.

```
def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
```

```
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

- Pour plus de détails sur l'API, consultez le DetachRoledocument de référence de l'API Policy in AWS SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, attache et détache les politiques de rôle.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
```

```
 # @return [String] The policy ARN if successful, otherwise nil
 def create_policy(policy_name, policy_document)
   response = @iam_client.create_policy(
     policy_name: policy_name,
     policy_document: policy_document.to_json
   )
   response.policy.arn
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error creating policy: #{e.message}")
   nil
 end

 # Fetches an IAM policy by its ARN
 # @param policy_arn [String] the ARN of the IAM policy to retrieve
 # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
 def get_policy(policy_arn)
   response = @iam_client.get_policy(policy_arn: policy_arn)
   policy = response.policy
   @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
   policy
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
   raise
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
   raise
 end

 # Attaches a policy to a role
 #
 # @param role_name [String] The name of the role
 # @param policy_arn [String] The policy ARN
 # @return [Boolean] true if successful, false otherwise
 def attach_policy_to_role(role_name, policy_arn)
   @iam_client.attach_role_policy(
     role_name: role_name,
     policy_arn: policy_arn
   )
   true
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error attaching policy to role: #{e.message}")
```

```
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, consultez DetachRolela section Politique dans le Guide de référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn detach_role_policy(
    client: &iamClient,
    role_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_role_policy()
        .role_name(role_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Pour plus de détails sur l'API, consultez DetachRolela section Politique du AWS SDK pour la référence de l'API Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d’être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   public func detachRolePolicy(policy: IAMClientTypes.Policy, role:
IAMClientTypes.Role) async throws {
       let input = DetachRolePolicyInput(
           policyArn: policy.arn,
           roleName: role.roleName
       )

       do {
           _ = try await iamClient.detachRolePolicy(input: input)
       } catch {
           throw error
       }
   }
```

- Pour plus de détails sur l'API, consultez DetachRolela section Politique du AWS SDK pour la référence à l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DetachUserPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DetachUserPolicy`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Créer des utilisateurs en lecture seule et en lecture-écriture

CLI

AWS CLI

Pour détacher une politique d'un utilisateur

Cet exemple supprime la politique gérée avec l'ARN `arn:aws:iam::123456789012:policy/TesterPolicy` de l'utilisateur Bob.

```
aws iam detach-user-policy \
    --user-name Bob \
    --policy-arn arn:aws:iam::123456789012:policy/TesterPolicy
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez Modification des autorisations pour un utilisateur IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez DetachUserla section Politique dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple détache la politique gérée dont l'ARN **`arn:aws:iam::123456789012:policy/TesterPolicy`** provient de l'utilisateur IAM nommé. **`Bob`**

```
Unregister-IAMUserPolicy -UserName Bob -PolicyArn
 arn:aws:iam::123456789012:policy/TesterPolicy
```

Exemple 2 : Cet exemple trouve toutes les politiques gérées associées à l'utilisateur IAM nommé **`Theresa`** et détache ces politiques de l'utilisateur.

```
Get-IAMAttachedUserPolicyList -UserName Theresa | Unregister-IAMUserPolicy -
Username Theresa
```

- Pour plus de détails sur l'API, consultez DetachUserla section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
        raise
```

- Pour plus de détails sur l'API, consultez le DetachUserdocument de référence de l'API Policy in AWS SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
  # Detaches a policy from a user
  #
  # @param user_name [String] The name of the user
  # @param policy_arn [String] The ARN of the policy to detach
  # @return [Boolean] true if the policy was successfully detached, false
otherwise
  def detach_user_policy(user_name, policy_arn)
    @iam_client.detach_user_policy(
      user_name: user_name,
      policy_arn: policy_arn
    )
    @logger.info("Policy '#{policy_arn}' detached from user '#{user_name}'
successfully.")
    true
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error detaching policy: Policy or user does not exist.")
    false
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from user '#{user_name}':
#{e.message}")
    false
  end
```

- Pour plus de détails sur l'API, consultez DetachUserla section Politique dans le Guide de référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn detach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
```

```
) -> Result<(), iamError> {
    client
        .detach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Pour plus de détails sur l'API, consultez DetachUserla section Politique du AWS SDK pour la référence de l'API Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`EnableMfaDevice`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`EnableMfaDevice`.

CLI

AWS CLI

Pour activer un appareil MFA

Après avoir utilisé la `create-virtual-mfa-device` commande pour créer un nouveau périphérique MFA virtuel, vous pouvez attribuer le périphérique MFA à un utilisateur. L'`enable-mfa-device`exemple suivant attribue le périphérique MFA avec le numéro de série à l'`arn:aws:iam::210987654321:mfa/BobsMFADevice`utilisateur. Bob La commande synchronise également le dispositif AWS en incluant les deux premiers codes en séquence à partir du dispositif MFA virtuel.

```
aws iam enable-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 789012
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Activation d'un dispositif d'authentification multifactorielle virtuelle (MFA) dans AWS le guide de l'utilisateur IAM.

- Pour plus de détails sur l'API, consultez EnableMfaDevice in AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cette commande active le périphérique MFA matériel avec le numéro de série **987654321098** et associe le périphérique à l'utilisateur. **Bob** Il inclut les deux premiers codes en séquence provenant de l'appareil.

```
Enable-IAMMFADevice -UserName "Bob" -SerialNumber "987654321098" -
AuthenticationCode1 "12345678" -AuthenticationCode2 "87654321"
```

Exemple 2 : Cet exemple crée et active un périphérique MFA virtuel. La première commande crée le périphérique virtuel et renvoie la représentation de l'objet du périphérique dans la variable**$MFADevice**. Vous pouvez utiliser les **QRCodePng** propriétés **.Base32StringSeed** or pour configurer l'application logicielle de l'utilisateur. La commande finale attribue l'appareil à l'utilisateur**David**, en identifiant l'appareil par son numéro de série. La commande synchronise également le dispositif AWS en incluant les deux premiers codes en séquence à partir du dispositif MFA virtuel.

```
$MFADevice = New-IAMVirtualMFADevice -VirtualMFADeviceName "MyMFADevice"
# see example for New-IAMVirtualMFADevice to see how to configure the software
 program with PNG or base32 seed code
Enable-IAMMFADevice -UserName "David" -SerialNumber -SerialNumber
 $MFADevice.SerialNumber -AuthenticationCode1 "24681357" -AuthenticationCode2
 "13572468"
```

- Pour plus de détails sur l'API, voir EnableMfaDevice in AWS Tools for PowerShell Cmdlet Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GenerateCredentialReport`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GenerateCredentialReport`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer votre compte

CLI

AWS CLI

Pour générer un rapport sur les informations d'identification

L'exemple suivant tente de générer un rapport d'identification pour le AWS compte.

```
aws iam generate-credential-report
```

Sortie :

```
{
    "State":  "STARTED",
    "Description": "No report exists. Starting a new report generation task"
}
```

Pour plus d'informations, consultez la section Obtenir des rapports d'identification pour votre AWS compte dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez GenerateCredentialle rapport dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple demande la génération d'un nouveau rapport, qui peut être effectué toutes les quatre heures. Si le dernier rapport est encore récent, le champ État est indiqué**COMPLETE**. **Get-IAMCredentialReport**À utiliser pour afficher le rapport complet.

```
Request-IAMCredentialReport
```

Sortie :

```
Description                                                  State
-----------                                                  -----
No report exists. Starting a new report generation task      STARTED
```

- Pour plus de détails sur l'API, consultez la section GenerateCredentialRapport dans la référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def generate_credential_report():
    """
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
        response = iam.meta.client.generate_credential_report()
        logger.info(
```

```
            "Generating credentials report for your account. " "Current state is
%s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
account.")
        raise
    else:
        return response
```

- Pour plus de détails sur l'API, reportez-vous à la section GenerateCredentialReport in AWS SDK for Python (Boto3) API Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GenerateServiceLastAccessedDetails`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GenerateServiceLastAccessedDetails`.

CLI

AWS CLI

Exemple 1 : pour générer un rapport d'accès aux services pour une politique personnalisée

L'`generate-service-last-accessed-details`exemple suivant lance une tâche en arrière-plan pour générer un rapport répertoriant les services auxquels accèdent les utilisateurs IAM et les autres entités avec une politique personnalisée nommée`intern-boundary`. Vous pouvez afficher le rapport une fois qu'il a été créé en exécutant la `get-service-last-accessed-details` commande.

```
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::123456789012:policy/intern-boundary
```

Sortie :

```
{
    "JobId": "2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc"
}
```

Exemple 2 : pour générer un rapport d'accès aux services pour la AdministratorAccess politique AWS gérée

L'`generate-service-last-accessed-details`exemple suivant lance une tâche en arrière-plan pour générer un rapport répertoriant les services auxquels accèdent les utilisateurs IAM et les autres entités avec la `AdministratorAccess` politique AWS gérée. Vous pouvez afficher le rapport une fois qu'il a été créé en exécutant la `get-service-last-accessed-details` commande.

```
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::aws:policy/AdministratorAccess
```

Sortie :

```
{
    "JobId": "78b6c2ba-d09e-6xmp-7039-ecde30b26916"
}
```

Pour plus d'informations, voir Affiner les autorisations lors de AWS l'utilisation des dernières informations consultées dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir GenerateServiceLastAccessedDétails dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple est une applet de commande équivalente à l' GenerateServiceLastAccessedDetails API. Cela fournit un identifiant de tâche qui peut être utilisé dans Get-IAM et Get-IAM ServiceLastAccessedDetail ServiceLast AccessedDetail WithEntity

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

- Pour plus de détails sur l'API, reportez-vous à la section GenerateServiceLastAccessedDétails de la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetAccessKeyLastUsed`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetAccessKeyLastUsed`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer des clés d'accès

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::accessKeyLastUsed(const Aws::String &secretKeyID,
                                    const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetAccessKeyLastUsedRequest request;

    request.SetAccessKeyId(secretKeyID);

    Aws::IAM::Model::GetAccessKeyLastUsedOutcome outcome =
 iam.GetAccessKeyLastUsed(
            request);
```

```
    if (!outcome.IsSuccess()) {
        std::cerr << "Error querying last used time for access key " <<
                  secretKeyID << ":" << outcome.GetError().GetMessage() <<
 std::endl;
    }
    else {
        Aws::String lastUsedTimeString =
                outcome.GetResult()
                        .GetAccessKeyLastUsed()
                        .GetLastUsedDate()
                        .ToGmtString(Aws::Utils::DateFormat::ISO_8601);
        std::cout << "Access key " << secretKeyID << " last used at time " <<
                  lastUsedTimeString << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, voir GetAccessKeyLastUtilisé dans les références AWS SDK for C++ d'API.

CLI

AWS CLI

Pour récupérer des informations relatives au moment où la clé d’accès spécifiée a été utilisée pour la dernière fois

L’exemple suivant récupère des informations relatives au moment où la clé d’accès ABCDEXAMPLE a été utilisée pour la dernière fois.

```
aws iam get-access-key-last-used \
    --access-key-id ABCDEXAMPLE
```

Sortie :

```
{
    "UserName":  "Bob",
    "AccessKeyLastUsed": {
```

```
        "Region": "us-east-1",
        "ServiceName": "iam",
        "LastUsedDate": "2015-06-16T22:45:00Z"
    }
}
```

Pour de plus amples informations, veuillez consulter Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section GetAccessKeyLastUtilisée dans le AWS CLI manuel de référence des commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Obtenez la clé d'accès.

```
import { GetAccessKeyLastUsedCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} accessKeyId
 */
export const getAccessKeyLastUsed = async (accessKeyId) => {
  const command = new GetAccessKeyLastUsedCommand({
    AccessKeyId: accessKeyId,
  });

  const response = await client.send(command);

  if (response.AccessKeyLastUsed?.LastUsedDate) {
    console.log(`
    ${accessKeyId} was last used by ${response.UserName} via
```

```
    the ${response.AccessKeyLastUsed.ServiceName} service on
    ${response.AccessKeyLastUsed.LastUsedDate.toISOString()}
    `);
  }

  return response;
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, voir GetAccessKeyLastUtilisé dans les références AWS SDK for JavaScript d'API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.getAccessKeyLastUsed(
  { AccessKeyId: "ACCESS_KEY_ID" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data.AccessKeyLastUsed);
    }
  }
);
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, voir GetAccessKeyLastUtilisé dans les références AWS SDK for JavaScript d'API.

PowerShell

Outils pour PowerShell

Exemple 1 : renvoie le nom d'utilisateur propriétaire et les informations relatives à la dernière utilisation de la clé d'accès fournie.

```
Get-IAMAccessKeyLastUsed -AccessKeyId ABCDEXAMPLE
```

- Pour plus de détails sur l'API, consultez la section GetAccessKeyLastUtilisation dans la référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.

    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
```

```
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response
```

- Pour plus de détails sur l'API, consultez le GetAccessKeyLastmanuel de référence de l'API Used in AWS SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetAccountAuthorizationDetails`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetAccountAuthorizationDetails`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer votre compte

CLI

AWS CLI

Pour répertorier les utilisateurs, les groupes, les rôles et les politiques IAM d'un AWS compte

La `get-account-authorization-details` commande suivante renvoie des informations sur tous les utilisateurs, groupes, rôles et politiques IAM du AWS compte.

```
aws iam get-account-authorization-details
```

Sortie :

```
{
    "RoleDetailList": [
        {
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "RoleId": "AROA1234567890EXAMPLE",
            "CreateDate": "2014-07-30T17:09:20Z",
            "InstanceProfileList": [
                {
                    "InstanceProfileId": "AIPA1234567890EXAMPLE",
                    "Roles": [
                        {
                            "AssumeRolePolicyDocument": {
                                "Version": "2012-10-17",
                                "Statement": [
                                    {
                                        "Sid": "",
                                        "Effect": "Allow",
                                        "Principal": {
                                            "Service": "ec2.amazonaws.com"
                                        },
                                        "Action": "sts:AssumeRole"
                                    }
                                ]
                            },
                            "RoleId": "AROA1234567890EXAMPLE",
                            "CreateDate": "2014-07-30T17:09:20Z",
                            "RoleName": "EC2role",
```

```
                            "Path": "/",
                            "Arn": "arn:aws:iam::123456789012:role/EC2role"
                        }
                    ],
                    "CreateDate": "2014-07-30T17:09:20Z",
                    "InstanceProfileName": "EC2role",
                    "Path": "/",
                    "Arn": "arn:aws:iam::123456789012:instance-profile/EC2role"
                }
            ],
            "RoleName": "EC2role",
            "Path": "/",
            "AttachedManagedPolicies": [
                {
                    "PolicyName": "AmazonS3FullAccess",
                    "PolicyArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
                },
                {
                    "PolicyName": "AmazonDynamoDBFullAccess",
                    "PolicyArn": "arn:aws:iam::aws:policy/
AmazonDynamoDBFullAccess"
                }
            ],
            "RoleLastUsed": {
                "Region": "us-west-2",
                "LastUsedDate": "2019-11-13T17:30:00Z"
            },
            "RolePolicyList": [],
            "Arn": "arn:aws:iam::123456789012:role/EC2role"
        }
    ],
    "GroupDetailList": [
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "AdministratorAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
            },
            "GroupName": "Admins",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "CreateDate": "2013-10-14T18:32:24Z",
            "GroupPolicyList": []
        },
```

```
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "PowerUserAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/PowerUserAccess"
            },
            "GroupName": "Dev",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Dev",
            "CreateDate": "2013-10-14T18:33:55Z",
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": [],
            "GroupName": "Finance",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Finance",
            "CreateDate": "2013-10-14T18:57:48Z",
            "GroupPolicyList": [
                {
                    "PolicyName": "policygen-201310141157",
                    "PolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Action": "aws-portal:*",
                                "Sid": "Stmt1381777017000",
                                "Resource": "*",
                                "Effect": "Allow"
                            }
                        ]
                    }
                }
            ]
        }
    ],
    "UserDetailList": [
        {
            "UserName": "Alice",
            "GroupList": [
                "Admins"
            ],
            "CreateDate": "2013-10-14T18:32:24Z",
```

```
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Alice"
        },
        {
            "UserName": "Bob",
            "GroupList": [
                "Admins"
            ],
            "CreateDate": "2013-10-14T18:32:25Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [
                {
                    "PolicyName": "DenyBillingAndIAMPolicy",
                    "PolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": {
                            "Effect": "Deny",
                            "Action": [
                                "aws-portal:*",
                                "iam:*"
                            ],
                            "Resource": "*"
                        }
                    }
                }
            ],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Bob"
        },
        {
            "UserName": "Charlie",
            "GroupList": [
                "Dev"
            ],
            "CreateDate": "2013-10-14T18:33:56Z",
            "UserId": "AIDA1234567890EXAMPLE",
            "UserPolicyList": [],
            "Path": "/",
            "AttachedManagedPolicies": [],
            "Arn": "arn:aws:iam::123456789012:user/Charlie"
```

```
        }
    ],
    "Policies": [
        {
            "PolicyName": "create-update-delete-set-managed-policies",
            "CreateDate": "2015-02-06T19:58:34Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2015-02-06T19:58:34Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version": "2012-10-17",
                        "Statement": {
                            "Effect": "Allow",
                            "Action": [
                                "iam:CreatePolicy",
                                "iam:CreatePolicyVersion",
                                "iam:DeletePolicy",
                                "iam:DeletePolicyVersion",
                                "iam:GetPolicy",
                                "iam:GetPolicyVersion",
                                "iam:ListPolicies",
                                "iam:ListPolicyVersions",
                                "iam:SetDefaultPolicyVersion"
                            ],
                            "Resource": "*"
                        }
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/create-update-delete-set-
managed-policies",
            "UpdateDate": "2015-02-06T19:58:34Z"
        },
        {
            "PolicyName": "S3-read-only-specific-bucket",
            "CreateDate": "2015-01-21T21:39:41Z",
            "AttachmentCount": 1,
```

```
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2015-01-21T21:39:41Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "s3:Get*",
                                    "s3:List*"
                                ],
                                "Resource": [
                                    "arn:aws:s3:::example-bucket",
                                    "arn:aws:s3:::example-bucket/*"
                                ]
                            }
                        ]
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/S3-read-only-specific-
bucket",
            "UpdateDate": "2015-01-21T23:39:41Z"
        },
        {
            "PolicyName": "AmazonEC2FullAccess",
            "CreateDate": "2015-02-06T18:40:15Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2014-10-30T20:59:46Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version": "2012-10-17",
```

```
                        "Statement": [
                            {
                                "Action": "ec2:*",
                                "Effect": "Allow",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "elasticloadbalancing:*",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "cloudwatch:*",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "autoscaling:*",
                                "Resource": "*"
                            }
                        ]
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::aws:policy/AmazonEC2FullAccess",
            "UpdateDate": "2015-02-06T18:40:15Z"
        }
    ],
    "Marker": "EXAMPLEkakv9BCuUNFDtxWSyfzetYwEx2ADc8dnzfvERF5S6YMvXKx41t6gCl/
eeaCX3Jo94/bKqezEAg8TEVS99EKFLxm3jtbpl25FDWEXAMPLE",
    "IsTruncated": true
}
```

Pour plus d'informations, veuillez consulter Consignes pour les audits de sécuritéAWS dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetAccountAuthorizationDetailsà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple obtient les détails d'autorisation concernant les identités du AWS compte et affiche la liste des éléments de l'objet renvoyé, y compris les utilisateurs, les groupes et les rôles. Par exemple, la **`UserDetailList`** propriété affiche des informations sur les utilisateurs. Des informations similaires sont disponibles dans les **`GroupDetailList`** propriétés **`RoleDetailList`** et.

```
$Details=Get-IAMAccountAuthorizationDetail
$Details
```

Sortie :

```
GroupDetailList : {Administrators, Developers, Testers, Backup}
IsTruncated     : False
Marker          :
RoleDetailList  : {TestRole1, AdminRole, TesterRole, clirole...}
UserDetailList  : {Administrator, Bob, BackupToS3, }
```

```
$Details.UserDetailList
```

Sortie :

```
Arn            : arn:aws:iam::123456789012:user/Administrator
CreateDate     : 10/16/2014 9:03:09 AM
GroupList      : {Administrators}
Path           : /
UserId         : AIDACKCEVSQ6CEXAMPLE1
UserName       : Administrator
UserPolicyList : {}

Arn            : arn:aws:iam::123456789012:user/Bob
CreateDate     : 4/6/2015 12:54:42 PM
GroupList      : {Developers}
Path           : /
UserId         : AIDACKCEVSQ6CEXAMPLE2
UserName       : bab
UserPolicyList : {}
```

```
Arn             : arn:aws:iam::123456789012:user/BackupToS3
CreateDate      : 1/27/2015 10:15:08 AM
GroupList       : {Backup}
Path            : /
UserId          : AIDACKCEVSQ6CEXAMPLE3
UserName        : BackupToS3
UserPolicyList : {BackupServicePermissionsToS3Buckets}
```

- Pour plus de détails sur l'API, reportez-vous GetAccountAuthorizationDetailsà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

- Pour plus de détails sur l'API, consultez GetAccountAuthorizationDetailsle AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetAccountPasswordPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetAccountPasswordPolicy`.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }
```

- Pour plus de détails sur l'API, voir GetAccountPasswordPolicyla section Référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour afficher la politique de mot de passe du compte actuel

La commande `get-account-password-policy` suivante affiche les détails de la politique de mot de passe du compte actuel.

```
aws iam get-account-password-policy
```

Sortie :

```
{
    "PasswordPolicy": {
        "AllowUsersToChangePassword": false,
        "RequireLowercaseCharacters": false,
        "RequireUppercaseCharacters": false,
        "MinimumPasswordLength": 8,
        "RequireNumbers": true,
        "RequireSymbols": true
    }
}
```

Si aucune politique de mot de passe n'est définie pour le compte, la commande renvoie une erreur `NoSuchEntity`.

Pour plus d'informations, consultez la section Définition d'une politique de mot de passe du compte pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir GetAccountPasswordPolicyla section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy() (*types.PasswordPolicy,
 error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}
```

- Pour plus de détails sur l'API, voir GetAccountPasswordPolicyla section Référence des AWS SDK for Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Obtenez la politique de mot de passe du compte.

```
import {
  GetAccountPasswordPolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const getAccountPasswordPolicy = async () => {
  const command = new GetAccountPasswordPolicyCommand({});

  const response = await client.send(command);
  console.log(response.PasswordPolicy);
  return response;
};
```

- Pour plus de détails sur l'API, voir GetAccountPasswordPolicyla section Référence des AWS SDK for JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getAccountPasswordPolicy()
    {
        return $this->iamClient->getAccountPasswordPolicy();
    }
```

- Pour plus de détails sur l'API, voir GetAccountPasswordPolicyla section Référence des AWS SDK for PHP API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie des informations sur la politique de mot de passe du compte courant. Si aucune politique de mot de passe n'est définie pour le compte, la commande renvoie une **`NoSuchEntity`** erreur.

```
Get-IAMAccountPasswordPolicy
```

Sortie :

```
AllowUsersToChangePassword : True
ExpirePasswords            : True
HardExpiry                 : False
MaxPasswordAge             : 90
MinimumPasswordLength      : 8
PasswordReusePrevention    : 20
RequireLowercaseCharacters : True
RequireNumbers             : True
RequireSymbols             : False
RequireUppercaseCharacters : True
```

- Pour plus de détails sur l'API, consultez la section GetAccountPasswordPolicyRéférence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def print_password_policy():
    """
    Prints the password policy for the account.
    """
    try:
        pw_policy = iam.AccountPasswordPolicy()
        print("Current account password policy:")
        print(
            f"\tallow_users_to_change_password:
 {pw_policy.allow_users_to_change_password}"
        )
        print(f"\texpire_passwords: {pw_policy.expire_passwords}")
        print(f"\thard_expiry: {pw_policy.hard_expiry}")
        print(f"\tmax_password_age: {pw_policy.max_password_age}")
        print(f"\tminimum_password_length: {pw_policy.minimum_password_length}")
        print(f"\tpassword_reuse_prevention:
 {pw_policy.password_reuse_prevention}")
        print(
            f"\trequire_lowercase_characters:
 {pw_policy.require_lowercase_characters}"
        )
        print(f"\trequire_numbers: {pw_policy.require_numbers}")
        print(f"\trequire_symbols: {pw_policy.require_symbols}")
        print(
            f"\trequire_uppercase_characters:
 {pw_policy.require_uppercase_characters}"
        )
        printed = True
    except ClientError as error:
        if error.response["Error"]["Code"] == "NoSuchEntity":
            print("The account does not have a password policy set.")
        else:
            logger.exception("Couldn't get account password policy.")
            raise
    else:
        return printed
```

- Pour plus de détails sur l'API, consultez GetAccountPasswordPolicyle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# Class to manage IAM account password policies
class PasswordPolicyManager
  attr_accessor :iam_client, :logger

  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "IAMPolicyManager"
  end

  # Retrieves and logs the account password policy
  def print_account_password_policy
    begin
      response = @iam_client.get_account_password_policy
      @logger.info("The account password policy is:
 #{response.password_policy.to_h}")
    rescue Aws::IAM::Errors::NoSuchEntity
      @logger.info("The account does not have a password policy.")
    rescue Aws::Errors::ServiceError => e
      @logger.error("Couldn't print the account password policy. Error: #{e.code}
 - #{e.message}")
      raise
    end
  end
end
```

- Pour plus de détails sur l'API, voir GetAccountPasswordPolicyla section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn get_account_password_policy(
    client: &iamClient,
) -> Result<GetAccountPasswordPolicyOutput,
 SdkError<GetAccountPasswordPolicyError>> {
    let response = client.get_account_password_policy().send().await?;

    Ok(response)
}
```

- Pour plus de détails sur l'API, voir GetAccountPasswordPolicyla section de référence de l'API AWS SDK for Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetAccountSummary`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetAccountSummary`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Gérer votre compte

CLI

AWS CLI

Pour récupérer des informations sur l'utilisation des entités IAM et les quotas IAM dans le compte actuel

La commande `get-account-summary` suivante renvoie des informations sur l'utilisation actuelle des entités IAM et les quotas IAM actuels dans le compte.

```
aws iam get-account-summary
```

Sortie :

```
{
    "SummaryMap": {
        "UsersQuota": 5000,
        "GroupsQuota": 100,
        "InstanceProfiles": 6,
        "SigningCertificatesPerUserQuota": 2,
        "AccountAccessKeysPresent": 0,
        "RolesQuota": 250,
        "RolePolicySizeQuota": 10240,
        "AccountSigningCertificatesPresent": 0,
        "Users": 27,
        "ServerCertificatesQuota": 20,
        "ServerCertificates": 0,
        "AssumeRolePolicySizeQuota": 2048,
        "Groups": 7,
        "MFADevicesInUse": 1,
        "Roles": 3,
        "AccountMFAEnabled": 1,
        "MFADevices": 3,
        "GroupsPerUserQuota": 10,
        "GroupPolicySizeQuota": 5120,
        "InstanceProfilesQuota": 100,
        "AccessKeysPerUserQuota": 2,
        "Providers": 0,
        "UserPolicySizeQuota": 2048
    }
}
```

Pour plus d'informations sur les limites des entités, consultez les quotas IAM et AWS STS dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez le GetAccountrésumé dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie des informations sur l'utilisation actuelle des entités IAM et les quotas actuels des entités IAM dans le. Compte AWS

```
Get-IAMAccountSummary
```

Sortie :

```
Key                                Value
Users                              7
GroupPolicySizeQuota               5120
PolicyVersionsInUseQuota           10000
ServerCertificatesQuota            20
AccountSigningCertificatesPresent  0
AccountAccessKeysPresent           0
Groups                             3
UsersQuota                         5000
RolePolicySizeQuota                10240
UserPolicySizeQuota                2048
GroupsPerUserQuota                 10
AssumeRolePolicySizeQuota          2048
AttachedPoliciesPerGroupQuota      2
Roles                              9
VersionsPerPolicyQuota             5
GroupsQuota                        100
PolicySizeQuota                    5120
Policies                           5
RolesQuota                         250
ServerCertificates                 0
AttachedPoliciesPerRoleQuota       2
MFADevicesInUse                    2
PoliciesQuota                      1000
```

```
AccountMFAEnabled                 1
Providers                         2
InstanceProfilesQuota             100
MFADevices                        4
AccessKeysPerUserQuota            2
AttachedPoliciesPerUserQuota      2
SigningCertificatesPerUserQuota   2
PolicyVersionsInUse               4
InstanceProfiles                  1
...
```

- Pour plus de détails sur l'API, consultez le GetAccountrésumé dans le manuel de référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def get_summary():
    """
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
        raise
    else:
        return summary.summary_map
```

- Pour plus de détails sur l'API, consultez le GetAccountrésumé dans le AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetContextKeysForCustomPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetContextKeysForCustomPolicy`.

CLI

AWS CLI

Exemple 1 : pour répertorier les clés de contexte référencées par une ou plusieurs politiques JSON personnalisées fournies en paramètre sur la ligne de commande

La `get-context-keys-for-custom-policy` commande suivante analyse chaque politique fournie et répertorie les clés de contexte utilisées par ces politiques. Utilisez cette commande pour identifier les valeurs de clé de contexte que vous devez fournir pour utiliser correctement les commandes du simulateur de politiques `simulate-custom-policy` et`simulate-custom-policy`. Vous pouvez également récupérer la liste des clés de contexte utilisées par toutes les politiques associées à un utilisateur ou à un rôle IAM à l'aide de la `get-context-keys-for-custom-policy` commande. Les valeurs de paramètres commençant `file://` par indiquent à la commande de lire le fichier et d'utiliser le contenu comme valeur du paramètre au lieu du nom du fichier lui-même.

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/${aws:username}","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
```

Sortie :

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
```

```
    ]
}
```

Exemple 2 : pour répertorier les clés de contexte référencées par une ou plusieurs politiques JSON personnalisées fournies en entrée de fichier

La `get-context-keys-for-custom-policy` commande suivante est identique à l'exemple précédent, sauf que les politiques sont fournies dans un fichier plutôt que sous forme de paramètre. Comme la commande attend une liste de chaînes JSON, et non une liste de structures JSON, le fichier doit être structuré comme suit, bien que vous puissiez le réduire en une seule.

```
[
    "Policy1",
    "Policy2"
]
```

Ainsi, par exemple, un fichier contenant la politique de l'exemple précédent doit ressembler à ce qui suit. Vous devez éviter chaque guillemet intégré dans la chaîne de politique en le faisant précéder d'une « barre oblique inverse ».

```
[ "{\"Version\": \"2012-10-17\", \"Statement\": {\"Effect\": \"Allow
\", \"Action\": \"dynamodb:*\", \"Resource\": \"arn:aws:dynamodb:us-
west-2:128716708097:table/${aws:username}\", \"Condition\": {\"DateGreaterThan\":
 {\"aws:CurrentTime\": \"2015-08-16T12:00:00Z\"}}}}" ]
```

Ce fichier peut ensuite être soumis à la commande suivante.

```
aws iam get-context-keys-for-custom-policy \
    --policy-input-list file://policyfile.json
```

Sortie :

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Pour plus d'informations, consultez la section Utilisation du simulateur de politique IAM (AWS CLI et AWS API) dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetContextKeysForCustomPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple récupère toutes les clés de contexte présentes dans la politique JSON fournie. Afin de fournir plusieurs politiques, vous pouvez fournir une liste de valeurs séparées par des virgules.

```
$policy1 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
$policy2 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForCustomPolicy -PolicyInputList $policy1,$policy2
```

- Pour plus de détails sur l'API, reportez-vous GetContextKeysForCustomPolicyà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetContextKeysForPrincipalPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetContextKeysForPrincipalPolicy`.

CLI

AWS CLI

Pour répertorier les clés de contexte référencées par toutes les politiques associées à un principal IAM

La `get-context-keys-for-principal-policy` commande suivante permet de récupérer toutes les politiques associées à l'utilisateur `saanvi` et aux groupes dont elle est membre. Il analyse ensuite chacune d'elles et répertorie les clés de contexte utilisées par ces politiques. Utilisez cette commande pour identifier les valeurs de clé de contexte que vous devez fournir pour utiliser correctement `simulate-principal-policy` les commandes `simulate-custom-policy` et. Vous pouvez également récupérer la liste des clés de contexte utilisées par une politique JSON arbitraire à l'aide de la `get-context-keys-for-custom-policy` commande.

```
aws iam get-context-keys-for-principal-policy \
   --policy-source-arn arn:aws:iam::123456789012:user/saanvi
```

Sortie :

```
{
    "ContextKeyNames": [
        "aws:username",
        "aws:CurrentTime"
    ]
}
```

Pour plus d'informations, consultez la section Utilisation du simulateur de politique IAM (AWS CLI et AWS API) dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetContextKeysForPrincipalPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple récupère toutes les clés de contexte présentes dans le json de politique fourni et les politiques attachées à l'entité IAM (utilisateur/rôle, etc.). Pour : PolicyInputList vous pouvez fournir une liste de valeurs multiples sous forme de valeurs séparées par des virgules.

```
$policy1 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/","Condition":{"DateGreaterThan":
{"aws:CurrentTime":"2015-08-16T12:00:00Z"}}}}'
```

```
$policy2 = '{"Version":"2012-10-17","Statement":
{"Effect":"Allow","Action":"dynamodb:*","Resource":"arn:aws:dynamodb:us-
west-2:123456789012:table/"}}'
Get-IAMContextKeysForPrincipalPolicy -PolicyInputList $policy1,$policy2 -
PolicySourceArn arn:aws:iam::852640994763:user/TestUser
```

- Pour plus de détails sur l'API, consultez la section GetContextKeysForPrincipalPolicyRéférence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetCredentialReport`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetCredentialReport`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Gérer votre compte

CLI

AWS CLI

Pour obtenir un rapport sur les informations d’identification

Cet exemple ouvre le rapport renvoyé et le transmet au pipeline sous la forme d’un tableau de lignes de texte.

```
aws iam get-credential-report
```

Sortie :

```
{
    "GeneratedTime":  "2015-06-17T19:11:50Z",
    "ReportFormat": "text/csv"
```

```
}
```

Pour plus d'informations, consultez la section Obtenir des rapports d'identification pour votre AWS compte dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez GetCredentialle rapport dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple ouvre le rapport renvoyé et le transmet au pipeline sous forme de tableau de lignes de texte. La première ligne est l'en-tête avec les noms de colonnes séparés par des virgules. Chaque ligne successive est la ligne de détail d'un utilisateur, chaque champ étant séparé par des virgules. Avant de pouvoir consulter le rapport, vous devez le générer à l'aide de l'**Request-IAMCredentialReport**applet de commande. Pour récupérer le rapport sous forme de chaîne unique, utilisez à la **-Raw** place de**-AsTextArray**. L'alias **-SplitLines** est également accepté pour le **-AsTextArray** commutateur. Pour la liste complète des colonnes de la sortie, consultez la référence de l'API de service. Notez que si vous n'utilisez pas **-AsTextArray** ou**-SplitLines**, vous devez extraire le texte de la **.Content** propriété à l'aide de la **StreamReader** classe .NET.

```
Request-IAMCredentialReport
```

Sortie :

```
Description                                                          State
-----------                                                          -----
No report exists. Starting a new report generation task              STARTED
```

```
Get-IAMCredentialReport -AsTextArray
```

Sortie :

```
 user,arn,user_creation_time,password_enabled,password_last_used,password_last_changed,pa
 root_account,arn:aws:iam::123456789012:root,2014-10-15T16:31:25+00:00,not_supported,2015
A,false,N/A,false,N/A,false,N/A
```

```
Administrator,arn:aws:iam::123456789012:user/
Administrator,2014-10-16T16:03:09+00:00,true,2015-04-20T15:18:32+00:00,2014-10-16T16:06:0
A,false,true,2014-12-03T18:53:41+00:00,true,2015-03-25T20:38:14+00:00,false,N/
A,false,N/A
Bill,arn:aws:iam::123456789012:user/Bill,2015-04-15T18:27:44+00:00,false,N/A,N/
A,N/A,false,false,N/A,false,N/A,false,2015-04-20T20:00:12+00:00,false,N/A
```

- Pour plus de détails sur l'API, consultez la section GetCredentialRapport dans la référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]
```

- Pour plus de détails sur l'API, reportez-vous à la section GetCredentialReport in AWS SDK for Python (Boto3) API Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetGroup`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetGroup`.

CLI

AWS CLI

Pour obtenir un groupe IAM

Cet exemple renvoie des informations sur le groupe `Admins` IAM.

```
aws iam get-group \
    --group-name Admins
```

Sortie :

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-06-16T19:41:48Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    },
    "Users": []
}
```

Pour plus d'informations, consultez la section Identités IAM (utilisateurs, groupes d'utilisateurs et rôles) dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetGroupà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie des informations sur le groupe IAM**Testers**, y compris une collection de tous les utilisateurs IAM appartenant au groupe.

```
$results = Get-IAMGroup -GroupName "Testers"
$results
```

Sortie :

```
Group                                    IsTruncated           Marker
     Users
-----                                    -----------           ------
     -----
Amazon.IdentityManagement.Model.Group    False
     {Theresa, David}
```

```
$results.Group
```

Sortie :

```
Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : 3RHNZZGQJ7QHMAEXAMPLE1
GroupName  : Testers
Path       : /
```

```
$results.Users
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:user/Theresa
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 1/1/0001 12:00:00 AM
Path             : /
UserId           : 4OSVDDJJTF4XEEXAMPLE2
UserName         : Theresa
```

```
Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE3
UserName         : David
```

- Pour plus de détails sur l'API, reportez-vous GetGroupà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetGroupPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetGroupPolicy`.

CLI

AWS CLI

Pour obtenir des informations sur une politique attachée à un groupe IAM

La `get-group-policy` commande suivante permet d'obtenir des informations sur la politique spécifiée attachée au groupe nommé`Test-Group`.

```
aws iam get-group-policy \
    --group-name Test-Group \
    --policy-name S3-ReadOnly-Policy
```

Sortie :

```
{
    "GroupName": "Test-Group",
    "PolicyDocument": {
        "Statement": [
            {
                "Action": [
```

```
                    "s3:Get*",
                    "s3:List*"
                ],
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    },
    "PolicyName": "S3-ReadOnly-Policy"
}
```

Pour plus d'informations, consultez Gestion des politiques IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez GetGroupla section Politique dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie des informations sur la politique intégrée nommée **`PowerUserAccess-Testers`** pour le groupe**`Testers`**. La **`PolicyDocument`** propriété est codée en URL. Il est décodé dans cet exemple à l'aide de la méthode **`UrlDecode`** .NET.

```
$results = Get-IAMGroupPolicy -GroupName Testers -PolicyName PowerUserAccess-
Testers
$results
```

Sortie :

```
GroupName     PolicyDocument
 PolicyName
---------     --------------
 ----------
Testers       %7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%0A%20...
 PowerUserAccess-Testers

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "NotAction": "iam:*",
      "Resource": "*"
    }
  ]
}
```

- Pour plus de détails sur l'API, consultez GetGroupla section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetInstanceProfile`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetInstanceProfile`.

CLI

AWS CLI

Pour obtenir des informations sur le profil d'une instance

La `get-instance-profile` commande suivante permet d'obtenir des informations sur le profil d'instance nomméExampleInstanceProfile.

```
aws iam get-instance-profile \
    --instance-profile-name ExampleInstanceProfile
```

Sortie :

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AID2MAB8DPLSRHEXAMPLE",
        "Roles": [
            {
                "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
```

```
                "RoleId": "AIDGPMS9RO4H3FEXAMPLE",
                "CreateDate": "2013-01-09T06:33:26Z",
                "RoleName": "Test-Role",
                "Path": "/",
                "Arn": "arn:aws:iam::336924118301:role/Test-Role"
            }
        ],
        "CreateDate": "2013-06-12T23:52:02Z",
        "InstanceProfileName": "ExampleInstanceProfile",
        "Path": "/",
        "Arn": "arn:aws:iam::336924118301:instance-profile/
ExampleInstanceProfile"
    }
}
```

Pour de plus amples informations, consultez Utilisation de profils d'instance dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section GetInstanceProfil dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie les détails du profil d'instance nommé **ec2instancerole** défini dans le AWS compte courant.

```
Get-IAMInstanceProfile -InstanceProfileName ec2instancerole
```

Sortie :

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Pour plus de détails sur l'API, consultez la section GetInstanceProfil dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetLoginProfile`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetLoginProfile`.

CLI

AWS CLI

Pour obtenir des informations sur le mot de passe d'un utilisateur IAM

La `get-login-profile` commande suivante permet d'obtenir des informations sur le mot de passe de l'utilisateur IAM nomméBob.

```
aws iam get-login-profile \
    --user-name Bob
```

Sortie :

```
{
    "LoginProfile": {
        "UserName": "Bob",
        "CreateDate": "2012-09-21T23:03:39Z"
    }
}
```

La `get-login-profile` commande peut être utilisée pour vérifier qu'un utilisateur IAM possède un mot de passe. La commande renvoie une `NoSuchEntity` erreur si aucun mot de passe n'est défini pour l'utilisateur.

Vous ne pouvez pas afficher un mot de passe à l'aide de cette commande. Si le mot de passe est perdu, vous pouvez le réinitialiser (`update-login-profile`) pour l'utilisateur. Vous pouvez également supprimer le profil de connexion (`delete-login-profile`) de l'utilisateur, puis en créer un nouveau (`create-login-profile`).

Pour plus d'informations, consultez la section Gestion des mots de passe pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir GetLoginProfil dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie la date de création du mot de passe et indique si une réinitialisation du mot de passe est requise pour l'utilisateur **`David`** IAM.

```
Get-IAMLoginProfile -UserName David
```

Sortie :

```
CreateDate                 PasswordResetRequired              UserName
----------                 ---------------------              --------
12/10/2014 3:39:44 PM      False                              David
```

- Pour plus de détails sur l'API, consultez la section GetLoginProfil dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetOpenIdConnectProvider`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetOpenIdConnectProvider`.

CLI

AWS CLI

Pour renvoyer des informations sur le fournisseur OpenID Connect spécifié

Cet exemple renvoie des informations sur le fournisseur OpenID Connect dont l'ARN est. `arn:aws:iam::123456789012:oidc-provider/server.example.com`

```
aws iam get-open-id-connect-provider \
```

```
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
server.example.com
```

Sortie :

```
{
    "Url": "server.example.com"
        "CreateDate": "2015-06-16T19:41:48Z",
        "ThumbprintList": [
        "12345abcdefghijk67890lmnopqrst987example"
        ],
        "ClientIDList": [
        "example-application-ID"
        ]
}
```

Pour plus d'informations, consultez la section Création de fournisseurs d'identité OpenID Connect (OIDC) dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir GetOpenIdConnectProvider dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie des informations sur le fournisseur OpenID Connect dont l'ARN est. **`arn:aws:iam::123456789012:oidc-provider/accounts.google.com`** La **`ClientIDList`** propriété est une collection qui contient tous les ID clients définis pour ce fournisseur.

```
Get-IAMOpenIDConnectProvider -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/oidc.example.com
```

Sortie :

```
ClientIDList         CreateDate               ThumbprintList
           Url
------------         ----------               --------------
           ---
```

```
{MyOIDCApp}          2/3/2015 3:00:30 PM
 {12345abcdefghijk67890lmnopqrst98765uvwxy}   oidc.example.com
```

- Pour plus de détails sur l'API, consultez la section GetOpenIdConnectFournisseur dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetPolicy`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Utiliser l'API IAM Policy Builder

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {
```

```
        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
{ PolicyArn = policyArn });
        return response.Policy;
    }
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS SDK for .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::getPolicy(const Aws::String &policyArn,
                            const Aws::Client::ClientConfiguration &clientConfig)
 {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetPolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.GetPolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error getting policy " << policyArn << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &policy = outcome.GetResult().GetPolicy();
        std::cout << "Name: " << policy.GetPolicyName() << std::endl <<
                  "ID: " << policy.GetPolicyId() << std::endl << "Arn: " <<
                  policy.GetArn() << std::endl << "Description: " <<
                  policy.GetDescription() << std::endl << "CreateDate: " <<

 policy.GetCreateDate().ToGmtString(Aws::Utils::DateFormat::ISO_8601)
                  << std::endl;
```

```
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour récupérer des informations sur la politique gérée spécifiée

Cet exemple renvoie des détails sur la politique gérée dont l'ARN est `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam get-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Sortie :

```
{
    "Policy": {
        "PolicyName": "MySamplePolicy",
        "CreateDate": "2015-06-17T19:23;32Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "Z27SI6FQMGNQ2EXAMPLE1",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:policy/MySamplePolicy",
        "UpdateDate": "2015-06-17T19:23:32Z"
    }
}
```

Pour de plus amples informations, veuillez consulter Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(policyArn string) (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(context.TODO(), &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS SDK for Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Obtenez la politique.

```
import { GetPolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const getPolicy = (policyArn) => {
  const command = new GetPolicyCommand({
    PolicyArn: policyArn,
  });

  return client.send(command);
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  PolicyArn: "arn:aws:iam::aws:policy/AWSLambdaExecute",
};

iam.getPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.Policy.Description);
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS SDK for JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun getIAMPolicy(policyArnVal: String?) {
    val request =
        GetPolicyRequest {
            policyArn = policyArnVal
```

```
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.getPolicy(request)
        println("Successfully retrieved policy ${response.policy?.policyName}")
    }
}
```

- Pour plus de détails sur l'API, consultez GetPolicyla section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getPolicy($policyArn)
    {
        return $this->customWaiter(function () use ($policyArn) {
            return $this->iamClient->getPolicy(['PolicyArn' => $policyArn]);
        });
    }
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS SDK for PHP API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie des détails sur la politique gérée dont l'ARN est l'ARN**`arn:aws:iam::123456789012:policy/MySamplePolicy`**.

```
Get-IAMPolicy -PolicyArn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Sortie :

```
Arn              : arn:aws:iam::aws:policy/MySamplePolicy
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : MySamplePolicy
UpdateDate       : 2/6/2015 10:40:08 AM
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
```

```
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
policy_arn)
        raise
    else:
        return policy_statement
```

- Pour plus de détails sur l'API, consultez GetPolicyle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Fetches an IAM policy by its ARN
 # @param policy_arn [String] the ARN of the IAM policy to retrieve
 # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
 def get_policy(policy_arn)
   response = @iam_client.get_policy(policy_arn: policy_arn)
   policy = response.policy
   @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
   policy
```

```
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
   raise
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
   raise
 end
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section Référence des AWS SDK for Ruby API.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    public func getPolicy(arn: String) async throws -> IAMClientTypes.Policy {
        let input = GetPolicyInput(
            policyArn: arn
        )
        do {
            let output = try await client.getPolicy(input: input)
            guard let policy = output.policy else {
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
```

```
        } catch {
            throw error
        }
    }
```

- Pour plus de détails sur l'API, reportez-vous GetPolicyà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetPolicyVersion`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetPolicyVersion`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Gérer les politiques
- Utiliser l'API IAM Policy Builder

CLI

AWS CLI

Pour récupérer des informations sur la version spécifiée de la politique gérée spécifiée

Cet exemple renvoie le document de politique pour la version v2 de la politique dont l'ARN est `arn:aws:iam::123456789012:policy/MyManagedPolicy`.

```
aws iam get-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Sortie :

```
{
```

```
    "PolicyVersion": {
        "Document": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Action": "iam:*",
                    "Resource": "*"
                }
            ]
        },
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2023-04-11T00:22:54+00:00"
    }
}
```

Pour de plus amples informations, veuillez consulter Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir GetPolicyVersion dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie le document de politique correspondant à la **v2** version de la politique dont l'ARN est l'ARN**arn:aws:iam::123456789012:policy/MyManagedPolicy**. Le document de politique contenu dans la **Document** propriété est codé en URL et est décodé dans cet exemple à l'aide de la méthode **UrlDecode** .NET.

```
$results = Get-IAMPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy -VersionId v2
$results
```

Sortie :

```
CreateDate             Document
 IsDefaultVersion     VersionId
----------             --------
 ----------------     ---------
```

```
2/12/2015 9:39:53 AM   %7B%0A%20%20%22Version%22%3A%20%222012-10...    True
          v2

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
$policy = [System.Web.HttpUtility]::UrlDecode($results.Document)
$policy
{
  "Version": "2012-10-17",
  "Statement":
    {
      "Effect": "Allow",
      "Action": "*",
      "Resource": "*"
    }
}
```

- Pour plus de détails sur l'API, consultez GetPolicyla section Version dans la référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
```

```
            logger.info("Got default policy doc for %s.", policy.policy_name)
            logger.info(policy_doc)
        except ClientError:
            logger.exception("Couldn't get default policy statement for %s.",
    policy_arn)
            raise
        else:
            return policy_statement
```

- Pour plus de détails sur l'API, voir GetPolicyVersion in AWS SDK for Python (Boto3) API Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetRole`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetRole`.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
```

```
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }
```

- Pour plus de détails sur l'API, voir GetRolela section Référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour obtenir des informations sur un rôle IAM

La commande `get-role` suivante permet d'obtenir des informations sur le rôle nommé `Test-Role`.

```
aws iam get-role \
    --role-name Test-Role
```

Sortie :

```
{
    "Role": {
        "Description": "Test Role",
        "AssumeRolePolicyDocument":"<URL-encoded-JSON>",
        "MaxSessionDuration": 3600,
        "RoleId": "AROA1234567890EXAMPLE",
        "CreateDate": "2019-11-13T16:45:56Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "RoleLastUsed": {
            "Region": "us-east-1",
            "LastUsedDate": "2019-11-13T17:14:00Z"
        },
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

La commande affiche la politique d'approbation attachée au rôle. Pour répertorier les politiques des autorisations attachées à un rôle, utilisez la commande `list-role-policies`.

Pour plus d'informations, consultez Création de rôles IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir GetRolela section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(roleName string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(context.TODO(),
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Pour plus de détails sur l'API, voir GetRolela section Référence des AWS SDK for Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Obtenez le rôle.

```
import { GetRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const getRole = (roleName) => {
  const command = new GetRoleCommand({
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, voir GetRolela section Référence des AWS SDK for JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getRole($roleName)
    {
        return $this->customWaiter(function () use ($roleName) {
            return $this->iamClient->getRole(['RoleName' => $roleName]);
        });
    }
```

- Pour plus de détails sur l'API, voir GetRolela section Référence des AWS SDK for PHP API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie les détails du**`lamda_exec_role`**. Il inclut le document de politique de confiance qui précise qui peut assumer ce rôle. Le document de politique est codé en URL et peut être décodé à l'aide de la **`UrlDecode`** méthode .NET. Dans cet exemple, tous les espaces blancs de la politique d'origine avaient été supprimés avant d'être chargés dans la politique. Pour consulter les documents de politique d'autorisation qui déterminent ce que peut faire une personne assumant le rôle, utilisez les documents **`Get-IAMRolePolicy`** pour les politiques intégrées et **`Get-IAMPolicyVersion`** pour les politiques gérées associées.

```
$results = Get-IamRole -RoleName lambda_exec_role
$results | Format-List
```

Sortie :

```
Arn                      : arn:aws:iam::123456789012:role/lambda_exec_role
AssumeRolePolicyDocument : %7B%22Version%22%3A%222012-10-17%22%2C%22Statement
%22%3A%5B%7B%22Sid%22
                           %3A%22%22%2C%22Effect%22%3A%22Allow%22%2C%22Principal
%22%3A%7B%22Service
                           %22%3A%22lambda.amazonaws.com%22%7D%2C%22Action%22%3A
%22sts%3AAssumeRole
                           %22%7D%5D%7D
CreateDate               : 4/2/2015 9:16:11 AM
Path                     : /
RoleId                   : 2YBIKAIBHNKB4EXAMPLE1
RoleName                 : lambda_exec_role
```

```
$policy = [System.Web.HttpUtility]::UrlDecode($results.AssumeRolePolicyDocument)
$policy
```

Sortie :

```
{"Version":"2012-10-17","Statement":[{"Sid":"","Effect":"Allow","Principal":
{"Service":"lambda.amazonaws.com"},"Action":"sts:AssumeRole"}]}
```

- Pour plus de détails sur l'API, consultez la section GetRoleRéférence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def get_role(role_name):
    """
    Gets a role by name.

    :param role_name: The name of the role to retrieve.
    :return: The specified role.
```

```
    """
    try:
        role = iam.Role(role_name)
        role.load()  # calls GetRole to load attributes
        logger.info("Got role with arn %s.", role.arn)
    except ClientError:
        logger.exception("Couldn't get role named %s.", role_name)
        raise
    else:
        return role
```

- Pour plus de détails sur l'API, consultez GetRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# Gets data about a role.
#
# @param name [String] The name of the role to look up.
# @return [Aws::IAM::Role] The retrieved role.
def get_role(name)
  role = @iam_client.get_role({
                               role_name: name,
                             }).role
  puts("Got data for role '#{role.role_name}'. Its ARN is '#{role.arn}'.")
rescue Aws::Errors::ServiceError => e
  puts("Couldn't get data for role '#{name}' Here's why:")
  puts("\t#{e.code}: #{e.message}")
  raise
else
```

```
    role
  end
```

- Pour plus de détails sur l'API, voir GetRolela section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn get_role(
    client: &iamClient,
    role_name: String,
) -> Result<GetRoleOutput, SdkError<GetRoleError>> {
    let response = client.get_role().role_name(role_name).send().await?;
    Ok(response)
}
```

- Pour plus de détails sur l'API, voir GetRolela section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
public func getRole(name: String) async throws -> IAMClientTypes.Role {
    let input = GetRoleInput(
        roleName: name
    )
    do {
        let output = try await client.getRole(input: input)
        guard let role = output.role else {
            throw ServiceHandlerError.noSuchRole
        }
        return role
    } catch {
        throw error
    }
}
```

- Pour plus de détails sur l'API, reportez-vous GetRoleà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetRolePolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetRolePolicy`.

CLI

AWS CLI

Pour obtenir des informations sur une politique associée à un rôle IAM

La `get-role-policy` commande suivante permet d'obtenir des informations sur la politique spécifiée attachée au rôle nomméTest-Role.

```
aws iam get-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

Sortie :

```
{
  "RoleName": "Test-Role",
  "PolicyDocument": {
      "Statement": [
          {
              "Action": [
                  "s3:ListBucket",
                  "s3:Put*",
                  "s3:Get*",
                  "s3:*MultipartUpload*"
              ],
              "Resource": "*",
              "Effect": "Allow",
              "Sid": "1"
          }
      ]
  }
  "PolicyName": "ExamplePolicy"
}
```

Pour plus d'informations, consultez Création de rôles IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez GetRolela section Politique dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie le document de politique d'autorisations pour la politique nommée **`oneClick_lambda_exec_role_policy`** qui est intégrée au rôle **`lamda_exec_role`** IAM. Le document de politique qui en résulte est codé en URL. Il est décodé dans cet exemple à l'aide de la méthode **`UrlDecode`** .NET.

```
$results = Get-IAMRolePolicy -RoleName lambda_exec_role -PolicyName
 oneClick_lambda_exec_role_policy
$results
```

Sortie :

```
PolicyDocument                                          PolicyName
              UserName
--------------                                          ----------
              --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...
 oneClick_lambda_exec_role_policy     lambda_exec_role
```

```
[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
```

Sortie :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:*"
      ],
      "Resource": "arn:aws:logs:*:*:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::*"
      ]
    }
  ]
}
```

- Pour plus de détails sur l'API, consultez GetRolela section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetSamlProvider`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetSamlProvider`.

CLI

AWS CLI

Pour récupérer le méta-document du fournisseur SAML

Cet exemple récupère les informations relatives au fournisseur SAML 2.0 dont l'ARM est le nom. `arn:aws:iam::123456789012:saml-provider/SAMLADFS` La réponse inclut le document de métadonnées que vous avez obtenu du fournisseur d'identité pour créer l'entité du fournisseur AWS SAML ainsi que les dates de création et d'expiration.

```
aws iam get-saml-provider \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

Sortie :

```
{
    "SAMLMetadataDocument": "...SAMLMetadataDocument-XML...",
    "CreateDate": "2017-03-06T22:29:46+00:00",
    "ValidUntil": "2117-03-06T22:29:46.433000+00:00",
    "Tags": [
        {
            "Key": "DeptID",
            "Value": "123456"
        },
        {
            "Key": "Department",
            "Value": "Accounting"
        }
```

```
    ]
}
```

Pour plus d'informations, consultez Création de fournisseurs d'identité SAML IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir GetSamlProvider dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple récupère les informations relatives au fournisseur SAML 2.0 dont l'ARM est arn:aws:iam : :123456789012:SAML-provider/samladfs. La réponse inclut le document de métadonnées que vous avez obtenu du fournisseur d'identité pour créer l'entité du fournisseur AWS SAML ainsi que les dates de création et d'expiration.

```
Get-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS
```

Sortie :

```
CreateDate                 SAMLMetadataDocument
        ValidUntil
----------                 --------------------
        ----------
12/23/2014 12:16:55 PM    <EntityDescriptor ID="_12345678-1234-5678-9012-
example1...    12/23/2114 12:16:54 PM
```

- Pour plus de détails sur l'API, consultez la section GetSamlFournisseur dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetServerCertificate`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetServerCertificate`.

C++

Kit de développement logiciel (SDK) for C++

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::getServerCertificate(const Aws::String &certificateName,
                                       const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetServerCertificateRequest request;
    request.SetServerCertificateName(certificateName);

    auto outcome = iam.GetServerCertificate(request);
    bool result = true;
    if (!outcome.IsSuccess()) {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error getting server certificate " << certificateName
 <<
                      ": " << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << certificateName
                      << "' not found." << std::endl;
        }
    }
    else {
        const auto &certificate = outcome.GetResult().GetServerCertificate();
        std::cout << "Name: " <<

 certificate.GetServerCertificateMetadata().GetServerCertificateName()
                  << std::endl << "Body: " << certificate.GetCertificateBody() <<
                  std::endl << "Chain: " << certificate.GetCertificateChain() <<
                  std::endl;
    }
```

```
    return result;
}
```

- Pour plus de détails sur l'API, consultez la section GetServerCertificat dans la référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour obtenir des informations sur un certificat de serveur associé à votre AWS compte

La `get-server-certificate` commande suivante permet de récupérer tous les détails relatifs au certificat de serveur spécifié dans votre AWS compte.

```
aws iam get-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

Sortie :

```
{
    "ServerCertificate": {
        "ServerCertificateMetadata": {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        "CertificateBody": "-----BEGIN CERTIFICATE-----
            MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
            VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
            b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
            BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
            MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
            VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
            b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
            YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
```

```
            21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
            rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
            Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
            nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
            FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
            NYiytVbZPQUQ5Yaxu2jXnimvrszlaEXAMPLE=-----END CERTIFICATE-----",
        "CertificateChain": "-----BEGIN CERTIFICATE-----\nMIICiTCCAfICCQD6md
            7oRw0uXOjANBgkqhkiG9w0BAqQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgT
            AldBMRAwDgYDVQQHEwdTZWF0drGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAs
            TC0lBTSBDb25zb2xlMRIwEAYDVsQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQ
            jb20wHhcNMTEwNDI1MjA0NTIxWhtcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBh
            MCVVMxCzAJBgNVBAgTAldBMRAwDgsYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBb
            WF6b24xFDASBgNVBAsTC0lBTSBDb2d5zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMx
            HzAdBgkqhkiG9w0BCQEWEG5vb25lQGfFtYXpvbi5jb20wgZ8wDQYJKoZIhvcNAQE
            BBQADgY0AMIGJAoGBAMaK0dn+a4GmWIgWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVI
            k60CpiwsZ3G93vUEIO3IyNoH/f0wYK8mh9TrDHudUZg3qX4waLG5M43q7Wgc/MbQ
            ITxOUSQv7c7ugFFDzQGBzZswY6786m86gjpEIbb3OhjZnzcvQAaRHhdlQWIMm2nr
            AgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCku4nUhVVxYUntneD9+h8Mg9q6q+auN
            KyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FlkbFFBjvSfpJIlJ00zbhNYS5f6Guo
            EDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjS;TbNYiytVbZPQUQ5Yaxu2jXnimvw
            3rrszlaEWEG5vb25lQGFtsYXpvbiEXAMPLE=\n-----END CERTIFICATE-----"
    }
}
```

Pour répertorier les certificats de serveur disponibles dans votre AWS compte, utilisez la `list-server-certificates` commande.

Pour de plus amples informations, veuillez consulter Gestion des certificats de serveur dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section GetServerCertificat dans AWS CLI la référence des commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Obtenez un certificat de serveur.

```
import { GetServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 * @returns
 */
export const getServerCertificate = async (certName) => {
  const command = new GetServerCertificateCommand({
    ServerCertificateName: certName,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez la section GetServerCertificat dans la référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });
```

```
iam.getServerCertificate(
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez la section GetServerCertificat dans la référence des AWS SDK for JavaScript API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple permet de récupérer des informations sur le certificat de serveur nommé**`MyServerCertificate`**. Vous trouverez les détails du certificat dans les **`ServerCertificateMetadata`** propriétés **`CertificateBody`** et.

```
$result = Get-IAMServerCertificate -ServerCertificateName MyServerCertificate
$result | format-list
```

Sortie :

```
CertificateBody          : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
```

```
 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt

 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
                            21uUSfwfEvySWtC2XADZ4nB
+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
                            rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                            nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                            NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                            -----END CERTIFICATE-----
CertificateChain          :
ServerCertificateMetadata :
 Amazon.IdentityManagement.Model.ServerCertificateMetadata
```

```
$result.ServerCertificateMetadata
```

Sortie :

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Pour plus de détails sur l'API, consultez la section GetServerCertificat dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetServiceLastAccessedDetails`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetServiceLastAccessedDetails`.

CLI

AWS CLI

Pour récupérer un rapport d'accès aux services

L'`get-service-last-accessed-details`exemple suivant récupère un rapport généré précédemment qui répertorie les services auxquels accèdent les entités IAM. Pour générer un rapport, utilisez la `generate-service-last-accessed-details` commande.

```
aws iam get-service-last-accessed-details \
    --job-id 2eb6c2b8-7b4c-3xmp-3c13-03b72c8cdfdc
```

Sortie :

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:50:35.929Z",
    "ServicesLastAccessed": [
        ...
        {
            "ServiceName": "AWS Lambda",
            "LastAuthenticated": "2019-09-30T23:02:00Z",
            "ServiceNamespace": "lambda",
            "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/admin",
            "TotalAuthenticatedEntities": 6
        },
    ]
}
```

Pour plus d'informations, voir Affiner les autorisations lors de AWS l'utilisation des dernières informations consultées dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous à la section GetServiceLastAccessedDétails dans le AWS CLI manuel de référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple fournit les détails du dernier accès au service par l'entité IAM (utilisateur, groupe, rôle ou politique) associée à l'appel de demande.

```
Request-IAMServiceLastAccessedDetail -Arn arn:aws:iam::123456789012:user/TestUser
```

Sortie :

```
f0b7a819-eab0-929b-dc26-ca598911cb9f
```

```
Get-IAMServiceLastAccessedDetail -JobId f0b7a819-eab0-929b-dc26-ca598911cb9f
```

- Pour plus de détails sur l'API, reportez-vous à la section GetServiceLastAccessedDétails de la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetServiceLastAccessedDetailsWithEntities`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetServiceLastAccessedDetailsWithEntities`.

CLI

AWS CLI

Pour récupérer un rapport d'accès aux services contenant les détails d'un service

L'`get-service-last-accessed-details-with-entities`exemple suivant extrait un rapport contenant des informations détaillées sur les utilisateurs IAM et les autres entités ayant

accédé au service spécifié. Pour générer un rapport, utilisez la `generate-service-last-accessed-details` commande. Pour obtenir la liste des services accessibles avec des espaces de noms, utilisez`get-service-last-accessed-details`.

```
aws iam get-service-last-accessed-details-with-entities \
    --job-id 78b6c2ba-d09e-6xmp-7039-ecde30b26916 \
    --service-namespace lambda
```

Sortie :

```
{
    "JobStatus": "COMPLETED",
    "JobCreationDate": "2019-10-01T03:55:41.756Z",
    "JobCompletionDate": "2019-10-01T03:55:42.533Z",
    "EntityDetailsList": [
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/admin",
                "Name": "admin",
                "Type": "USER",
                "Id": "AIDAIO2XMPLENQEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-30T23:02:00Z"
        },
        {
            "EntityInfo": {
                "Arn": "arn:aws:iam::123456789012:user/developer",
                "Name": "developer",
                "Type": "USER",
                "Id": "AIDAIBEYXMPL2YEXAMPLE",
                "Path": "/"
            },
            "LastAuthenticated": "2019-09-16T19:34:00Z"
        }
    ]
}
```

Pour plus d'informations, voir Affiner les autorisations lors de AWS l'utilisation des dernières informations consultées dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir GetServiceLastAccessedDetailsWithEntités dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple fournit l'horodatage du dernier accès au service dans la demande par cette entité IAM respective.

```
$results = Get-IAMServiceLastAccessedDetailWithEntity -JobId f0b7a819-eab0-929b-
dc26-ca598911cb9f -ServiceNamespace ec2
$results
```

Sortie :

```
EntityDetailsList : {Amazon.IdentityManagement.Model.EntityDetails}
Error             :
IsTruncated       : False
JobCompletionDate : 12/29/19 11:19:31 AM
JobCreationDate   : 12/29/19 11:19:31 AM
JobStatus         : COMPLETED
Marker            :
```

```
$results.EntityDetailsList
```

Sortie :

```
EntityInfo                                  LastAuthenticated
----------                                  -----------------
Amazon.IdentityManagement.Model.EntityInfo 11/16/19 3:47:00 PM
```

```
$results.EntityInfo
```

Sortie :

```
Arn  : arn:aws:iam::123456789012:user/TestUser
Id   : AIDA4NBK5CXF5TZHU1234
Name : TestUser
Path : /
```

```
Type : USER
```

- Pour plus de détails sur l'API, consultez GetServiceLastAccessedDetailsWithEntities in AWS Tools for PowerShell Cmdlet Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetServiceLinkedRoleDeletionStatus`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetServiceLinkedRoleDeletionStatus`.

CLI

AWS CLI

Pour vérifier le statut d'une requête de suppression d'un rôle lié à un service

L'exemple `get-service-linked-role-deletion-status` suivant affiche le statut d'une requête précédente de suppression d'un rôle lié à un service. L'opération de suppression s'effectue de manière asynchrone. Lorsque vous effectuez la requête, vous obtenez une valeur `DeletionTaskId` que vous fournissez en tant que paramètre de cette commande.

```
aws iam get-service-linked-role-deletion-status \
    --deletion-task-id task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE
```

Sortie :

```
{
"Status": "SUCCEEDED"
}
```

Pour plus d'informations, consultez Utilisation des rôles liés à un service dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetServiceLinkedRoleDeletionStatusà la section Référence des AWS CLI commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import {
  GetServiceLinkedRoleDeletionStatusCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} deletionTaskId
 */
export const getServiceLinkedRoleDeletionStatus = (deletionTaskId) => {
  const command = new GetServiceLinkedRoleDeletionStatusCommand({
    DeletionTaskId: deletionTaskId,
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, reportez-vous GetServiceLinkedRoleDeletionStatusà la section Référence des AWS SDK for JavaScript API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetUser`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetUser`.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }
```

- Pour plus de détails sur l'API, reportez-vous GetUserà la section Référence des AWS SDK for .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
```

```
# This function outputs everything sent to it to STDERR (standard error output).
########################################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

########################################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
########################################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}
```

- Pour plus de détails sur l'API, reportez-vous GetUserà la section Référence des AWS CLI commandes.

CLI

AWS CLI

Pour obtenir des informations sur un utilisateur IAM

La commande `get-user` suivante permet d'obtenir des informations sur l'utilisateur IAM nommé `Paulo`.

```
aws iam get-user \
    --user-name Paulo
```

Sortie :

```
{
    "User": {
        "UserName": "Paulo",
        "Path": "/",
        "CreateDate": "2019-09-21T23:03:13Z",
        "UserId": "AIDA123456789EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Paulo"
    }
}
```

Pour plus d'informations, consultez la section Gestion des utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous GetUserà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(context.TODO(), &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}
```

- Pour plus de détails sur l'API, reportez-vous GetUserà la section Référence des AWS SDK for Go API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple permet de récupérer des informations sur l'utilisateur nomméDavid.

```
Get-IAMUser -UserName David
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:user/David
CreateDate       : 12/10/2014 3:39:27 PM
PasswordLastUsed : 3/19/2015 8:44:04 AM
Path             : /
UserId           : Y4FKWQCXTA52QEXAMPLE1
UserName         : David
```

Exemple 2 : Cet exemple permet de récupérer des informations sur l'utilisateur IAM actuellement connecté.

```
Get-IAMUser
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:user/Bob
CreateDate       : 10/16/2014 9:03:09 AM
PasswordLastUsed : 3/4/2015 12:12:33 PM
Path             : /
UserId           : 7K3GJEANSKZF2EXAMPLE2
UserName         : Bob
```

- Pour plus de détails sur l'API, reportez-vous GetUserà la section Référence des AWS Tools for PowerShell applets de commande.

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Retrieves a user's details
 #
 # @param user_name [String] The name of the user to retrieve
 # @return [Aws::IAM::Types::User, nil] The user object if found, or nil if an
error occurred
 def get_user(user_name)
   response = @iam_client.get_user(user_name: user_name)
   response.user
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("User '#{user_name}' not found.")
   nil
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error retrieving user '#{user_name}': #{e.message}")
   nil
 end
```

- Pour plus de détails sur l'API, reportez-vous GetUserà la section Référence des AWS SDK for Ruby API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetUserPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetUserPolicy`.

CLI

AWS CLI

Pour répertorier les détails de la politique d'un utilisateur IAM

La `get-user-policy` commande suivante répertorie les détails de la politique spécifiée attachée à l'utilisateur IAM nomméBob.

```
aws iam get-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

Sortie :

```
{
    "UserName": "Bob",
    "PolicyName": "ExamplePolicy",
    "PolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Action": "*",
                "Resource": "*",
                "Effect": "Allow"
            }
        ]
    }
}
```

Pour obtenir une liste des politiques d'un utilisateur IAM, utilisez la commande `list-user-policies`.

Pour de plus amples informations, veuillez consulter Policies and permissions in IAM (Stratégies et autorisations dans IAM) dans le AWS IAM Guide de l'utilisateur.

- Pour plus de détails sur l'API, consultez GetUserla section Politique dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple récupère les détails de la politique intégrée nommée **`Davids_IAM_Admin_Policy`** qui est incorporée dans le nom de l'utilisateur IAM. **`David`** Le document de politique est codé en URL.

```
$results = Get-IAMUserPolicy -PolicyName Davids_IAM_Admin_Policy -UserName David
$results
```

Sortie :

```
PolicyDocument                                                  PolicyName
       UserName
--------------                                                  ----------
       --------
%7B%0A%20%20%22Version%22%3A%20%222012-10-17%22%2C%...    Davids_IAM_Admin_Policy
       David

[System.Reflection.Assembly]::LoadWithPartialName("System.Web.HttpUtility")
[System.Web.HttpUtility]::UrlDecode($results.PolicyDocument)
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:*"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

- Pour plus de détails sur l'API, consultez GetUserla section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListAccessKeys`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListAccessKeys`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer des clés d'accès

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_list_access_keys
#
# This function lists the access keys for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#
```

```
# Returns:
#       access_key_ids
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_list_access_keys() {

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_access_keys"
    echo "Lists the AWS Identity and Access Management (IAM) access key IDs for
 the specified user."
    echo "  -u user_name   The name of the IAM user."
    echo ""
  }

  local user_name response
  local option OPTARG # Required to use getopts command in a function.
  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam list-access-keys \
    --user-name "$user_name" \
```

```
    --output text \
    --query 'AccessKeyMetadata[].AccessKeyId')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-access-keys operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, voir ListAccessKeys in AWS CLI Command Reference.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::listAccessKeys(const Aws::String &userName,
                                 const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccessKeysRequest request;
    request.SetUserName(userName);

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccessKeys(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list access keys for user " << userName
```

```
                    << ": " << outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(32) << "UserName" <<
                      std::setw(30) << "KeyID" << std::setw(20) << "Status" <<
                      std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &keys = outcome.GetResult().GetAccessKeyMetadata();
        const Aws::String DATE_FORMAT = "%Y-%m-%d";

        for (const auto &key: keys) {
            Aws::String statusString =
                    Aws::IAM::Model::StatusTypeMapper::GetNameForStatusType(
                            key.GetStatus());
            std::cout << std::left << std::setw(32) << key.GetUserName() <<
                      std::setw(30) << key.GetAccessKeyId() << std::setw(20) <<
                      statusString << std::setw(20) <<
                      key.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
 std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Pour plus de détails sur l'API, consultez la section ListAccessKeys in AWS SDK for C++ API Reference.

CLI

AWS CLI

Pour répertorier les ID de clés d'accès d'un utilisateur IAM

La commande `list-access-keys` suivante répertorie les ID des clés d'accès de l'utilisateur IAM nommé Bob.

```
aws iam list-access-keys \
    --user-name Bob
```

Sortie :

```
{
    "AccessKeyMetadata": [
        {
            "UserName": "Bob",
            "Status": "Active",
            "CreateDate": "2013-06-04T18:17:34Z",
            "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
        },
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CreateDate": "2013-06-06T20:42:26Z",
            "AccessKeyId": "AKIAI44QH8DHBEXAMPLE"
        }
    ]
}
```

Vous ne pouvez pas répertorier les clés d'accès secrètes des utilisateurs IAM. Si les clés d'accès secrètes sont perdues, vous devez créer de nouvelles clés d'accès à l'aide de la commande `create-access-keys`.

Pour de plus amples informations, veuillez consulter Gestion des clés d'accès pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir ListAccessKeys in AWS CLI Command Reference.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
 &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Pour plus de détails sur l'API, consultez la section ListAccessKeys in AWS SDK for Go API Reference.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.AccessKeyMetadata;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccessKeysRequest;
import software.amazon.awssdk.services.iam.model.ListAccessKeysResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListAccessKeys {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <userName>\s

                Where:
                    userName - The name of the user for which access keys are
 retrieved.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }
```

```
        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listKeys(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void listKeys(IamClient iam, String userName) {
        try {
            boolean done = false;
            String newMarker = null;

            while (!done) {
                ListAccessKeysResponse response;

                if (newMarker == null) {
                    ListAccessKeysRequest request =
ListAccessKeysRequest.builder()
                            .userName(userName)
                            .build();

                    response = iam.listAccessKeys(request);

                } else {
                    ListAccessKeysRequest request =
ListAccessKeysRequest.builder()
                            .userName(userName)
                            .marker(newMarker)
                            .build();

                    response = iam.listAccessKeys(request);
                }

                for (AccessKeyMetadata metadata : response.accessKeyMetadata()) {
                    System.out.format("Retrieved access key %s",
metadata.accessKeyId());
                }

                if (!response.isTruncated()) {
```

```
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, consultez la section ListAccessKeys in AWS SDK for Java 2.x API Reference.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les clés d'accès.

```
import { ListAccessKeysCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} userName
 */
```

```
export async function* listAccessKeys(userName) {
  const command = new ListAccessKeysCommand({
    MaxItems: 5,
    UserName: userName,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListAccessKeysCommandOutput |
 undefined}
   */
  let response = await client.send(command);

  while (response?.AccessKeyMetadata?.length) {
    for (const key of response.AccessKeyMetadata) {
      yield key;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAccessKeysCommand({
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez la section ListAccessKeys in AWS SDK for JavaScript API Reference.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 5,
  UserName: "IAM_USER_NAME",
};

iam.listAccessKeys(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez la section ListAccessKeys in AWS SDK for JavaScript API Reference.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun listKeys(userNameVal: String?) {
    val request =
        ListAccessKeysRequest {
```

```
            userName = userNameVal
        }
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccessKeys(request)
        response.accessKeyMetadata?.forEach { md ->
            println("Retrieved access key ${md.accessKeyId}")
        }
    }
}
```

- Pour plus de détails sur l'API, voir ListAccessKeys in AWS SDK for Kotlin API reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cette commande répertorie les clés d'accès de l'utilisateur IAM nommé**Bob**. Notez que vous ne pouvez pas répertorier les clés d'accès secrètes pour les utilisateurs IAM. Si les clés d'accès secrètes sont perdues, vous devez créer de nouvelles clés d'accès avec l'**New-IAMAccessKey**applet de commande.

```
Get-IAMAccessKey -UserName "Bob"
```

Sortie :

```
AccessKeyId               CreateDate                 Status
 UserName
-----------               ----------                 ------
 --------
AKIAIOSFODNN7EXAMPLE      12/3/2014 10:53:41 AM      Active             Bob
AKIAI44QH8DHBEXAMPLE      6/6/2013 8:42:26 PM        Inactive           Bob
```

- Pour plus de détails sur l'API, consultez la section ListAccessKeys in AWS Tools for PowerShell Cmdlet Reference.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys
```

- Pour plus de détails sur l'API, consultez le ListAccessmanuel de référence de l'API Keys in AWS SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, désactive et supprime les clés d'accès.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "AccessKeyManager"
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity => e
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
 #{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded => e
    @logger.error("Error creating access key: limit exceeded. Cannot create
 more.")
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
```

```
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: "Inactive"
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, consultez la section ListAccessKeys in AWS SDK for Ruby API Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListAccountAliases`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListAccountAliases`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer votre compte

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool
AwsDoc::IAM::listAccountAliases(const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccountAliasesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccountAliases(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list account aliases: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        const auto &aliases = outcome.GetResult().GetAccountAliases();
```

```
        if (!header) {
            if (aliases.size() == 0) {
                std::cout << "Account has no aliases" << std::endl;
                break;
            }
            std::cout << std::left << std::setw(32) << "Alias" << std::endl;
            header = true;
        }

        for (const auto &alias: aliases) {
            std::cout << std::left << std::setw(32) << alias << std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Pour plus de détails sur l'API, consultez la section ListAccountAlias dans la référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour répertorier des alias de compte

La commande `list-account-aliases` suivante répertorie les alias du compte actuel.

```
aws iam list-account-aliases
```

Sortie :

```
{
    "AccountAliases": [
```

```
    "mycompany"
    ]
}
```

Pour plus d'informations, consultez la section Votre identifiant de AWS compte et son alias dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir ListAccountAlias dans AWS CLI Command Reference.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccountAliasesResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListAccountAliases {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listAliases(iam);
        System.out.println("Done");
```

```
        iam.close();
    }

    public static void listAliases(IamClient iam) {
        try {
            ListAccountAliasesResponse response = iam.listAccountAliases();
            for (String alias : response.accountAliases()) {
                System.out.printf("Retrieved account alias %s", alias);
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, consultez la section ListAccountAlias dans la référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les alias de compte.

```
import { ListAccountAliasesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
```

```
 */
export async function* listAccountAliases() {
  const command = new ListAccountAliasesCommand({ MaxItems: 5 });

  let response = await client.send(command);

  while (response.AccountAliases?.length) {
    for (const alias of response.AccountAliases) {
      yield alias;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAccountAliasesCommand({
          Marker: response.Marker,
          MaxItems: 5,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez la section ListAccountAlias dans la référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });
```

```
// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listAccountAliases({ MaxItems: 10 }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez la section ListAccountAlias dans la référence des AWS SDK for JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun listAliases() {
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccountAliases(ListAccountAliasesRequest {})
        response.accountAliases?.forEach { alias ->
            println("Retrieved account alias $alias")
        }
    }
}
```

- Pour plus de détails sur l'API, consultez la section ListAccountAlias dans le AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell

Exemple 1 : Cette commande renvoie l'alias de compte pour le Compte AWS.

```
Get-IAMAccountAlias
```

Sortie :

```
ExampleCo
```

- Pour plus de détails sur l'API, consultez la section ListAccountAlias dans la référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
```

```
            logger.exception("Couldn't list aliases for your account.")
            raise
        else:
            return response["AccountAliases"]
```

- Pour plus de détails sur l'API, consultez la section ListAccountAliases in AWS SDK for Python (Boto3) API Reference.

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez, créez et supprimez des alias de compte.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info("Account aliases are:")
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
```

```
      @logger.info("No account aliases found.")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, consultez la section ListAccountAlias dans la référence des AWS SDK for Ruby API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListAttachedGroupPolicies`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListAttachedGroupPolicies`.

CLI

AWS CLI

Pour répertorier toutes les politiques gérées associées au groupe spécifié

Cet exemple renvoie les noms et les ARN des politiques gérées associées au groupe IAM nommé `Admins` dans le AWS compte.

```
aws iam list-attached-group-policies \
    --group-name Admins
```

Sortie :

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Pour de plus amples informations, veuillez consulter Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListAttachedGroupPoliciesà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cette commande renvoie les noms et les ARN des politiques gérées associées au groupe IAM nommé **`Admins`** dans le AWS compte. Pour voir la liste des politiques intégrées au groupe, utilisez la **`Get-IAMGroupPolicyList`** commande.

```
Get-IAMAttachedGroupPolicyList -GroupName "Admins"
```

Sortie :

```
PolicyArn                                                PolicyName
---------                                                ----------
arn:aws:iam::aws:policy/SecurityAudit                    SecurityAudit
arn:aws:iam::aws:policy/AdministratorAccess              AdministratorAccess
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedGroupPoliciesà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListAttachedRolePolicies`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListAttachedRolePolicies`.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
```

```
        var attachedRolePoliciesPaginator =
 _IAMService.Paginators.ListAttachedRolePolicies(new
 ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedRolePoliciesà la section Référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour répertorier toutes les politiques gérées qui sont attachées au rôle spécifié

Cette commande renvoie les noms et les ARN des politiques gérées associées au rôle IAM nommé `SecurityAuditRole` dans le AWS compte.

```
aws iam list-attached-role-policies \
    --role-name SecurityAuditRole
```

Sortie :

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Pour de plus amples informations, veuillez consulter Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListAttachedRolePoliciesà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(roleName string)
 ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
```

```
}
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedRolePoliciesà la section Référence des AWS SDK for Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les politiques qui sont attachées à un rôle.

```
import {
  ListAttachedRolePoliciesCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 * @param {string} roleName
 */
export async function* listAttachedRolePolicies(roleName) {
  const command = new ListAttachedRolePoliciesCommand({
    RoleName: roleName,
  });

  let response = await client.send(command);

  while (response.AttachedPolicies?.length) {
    for (const policy of response.AttachedPolicies) {
```

```
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAttachedRolePoliciesCommand({
          RoleName: roleName,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedRolePoliciesà la section Référence des AWS SDK for JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listAttachedRolePolicies($roleName, $pathPrefix = "", $marker
 = "", $maxItems = 0)
    {
        $listAttachRolePoliciesArguments = ['RoleName' => $roleName];
        if ($pathPrefix) {
            $listAttachRolePoliciesArguments['PathPrefix'] = $pathPrefix;
        }
        if ($marker) {
```

```
            $listAttachRolePoliciesArguments['Marker'] = $marker;
        }
        if ($maxItems) {
            $listAttachRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->iamClient-
>listAttachedRolePolicies($listAttachRolePoliciesArguments);
    }
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedRolePoliciesà la section Référence des AWS SDK for PHP API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cette commande renvoie les noms et les ARN des politiques gérées associées au rôle IAM nommé **`SecurityAuditRole`** dans le AWS compte. Pour voir la liste des politiques intégrées au rôle, utilisez la **`Get-IAMRolePolicyList`** commande.

```
Get-IAMAttachedRolePolicyList -RoleName "SecurityAuditRole"
```

Sortie :

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/SecurityAudit                     SecurityAudit
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedRolePoliciesà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_attached_policies(role_name):
    """
    Lists policies attached to a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.attached_policies.all():
            logger.info("Got policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't list attached policies for %s.", role_name)
        raise
```

- Pour plus de détails sur l'API, consultez ListAttachedRolePoliciesle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, attache et détache les politiques de rôle.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
 #{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
 exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
```

```
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedRolePoliciesà la section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn list_attached_role_policies(
    client: &iamClient,
    role_name: String,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListAttachedRolePoliciesOutput,
 SdkError<ListAttachedRolePoliciesError>> {
    let response = client
        .list_attached_role_policies()
        .role_name(role_name)
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Pour plus de détails sur l'API, voir ListAttachedRolePoliciesla section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   /// Returns a list of AWS Identity and Access Management (IAM) policies
   /// that are attached to the role.
   ///
   /// - Parameter role: The IAM role to return the policy list for.
   ///
   /// - Returns: An array of `IAMClientTypes.AttachedPolicy` objects
   ///   describing each managed policy that's attached to the role.
   public func listAttachedRolePolicies(role: String) async throws ->
[IAMClientTypes.AttachedPolicy] {
       var policyList: [IAMClientTypes.AttachedPolicy] = []
       var marker: String? = nil
       var isTruncated: Bool

       repeat {
           let input = ListAttachedRolePoliciesInput(
               marker: marker,
               roleName: role
           )
           let output = try await client.listAttachedRolePolicies(input: input)
```

```
            guard let attachedPolicies = output.attachedPolicies else {
                return policyList
            }

            for attachedPolicy in attachedPolicies {
                policyList.append(attachedPolicy)
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        return policyList
    }
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedRolePoliciesà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListAttachedUserPolicies`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListAttachedUserPolicies`.

CLI

AWS CLI

Pour répertorier toutes les politiques gérées associées à l'utilisateur spécifié

Cette commande renvoie les noms et les ARN des politiques gérées pour l'utilisateur IAM nommé Bob dans le AWS compte.

```
aws iam list-attached-user-policies \
    --user-name Bob
```

Sortie :

```
{
    "AttachedPolicies": [
```

```
        {
            "PolicyName": "AdministratorAccess",
            "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
        },
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Pour de plus amples informations, veuillez consulter Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListAttachedUserPoliciesà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cette commande renvoie les noms et les ARN des politiques gérées pour l'utilisateur IAM nommé **Bob** dans le AWS compte. Pour voir la liste des politiques intégrées intégrées à l'utilisateur IAM, utilisez la **Get-IAMUserPolicyList** commande.

```
Get-IAMAttachedUserPolicyList -UserName "Bob"
```

Sortie :

```
PolicyArn                                                 PolicyName
---------                                                 ----------
arn:aws:iam::aws:policy/TesterPolicy                      TesterPolicy
```

- Pour plus de détails sur l'API, reportez-vous ListAttachedUserPoliciesà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListEntitiesForPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListEntitiesForPolicy`.

CLI

AWS CLI

Pour répertorier tous les utilisateurs, groupes et rôles auxquels la politique gérée spécifiée est attachée

Cet exemple renvoie une liste des groupes, des rôles et des utilisateurs IAM auxquels la politique est `arn:aws:iam::123456789012:policy/TestPolicy` attachée.

```
aws iam list-entities-for-policy \
    --policy-arn arn:aws:iam::123456789012:policy/TestPolicy
```

Sortie :

```
{
    "PolicyGroups": [
        {
            "GroupName": "Admins",
            "GroupId": "AGPACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyUsers": [
        {
            "UserName": "Alice",
            "UserId": "AIDACKCEVSQ6C2EXAMPLE"
        }
    ],
    "PolicyRoles": [
        {
            "RoleName": "DevRole",
            "RoleId": "AROADBQP57FF2AEXAMPLE"
        }
    ],
    "IsTruncated": false
}
```

Pour de plus amples informations, veuillez consulter Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListEntitiesForPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie une liste de groupes, de rôles et d'utilisateurs IAM auxquels la politique est **`arn:aws:iam::123456789012:policy/TestPolicy`** attachée.

```
Get-IAMEntitiesForPolicy -PolicyArn "arn:aws:iam::123456789012:policy/TestPolicy"
```

Sortie :

```
IsTruncated  : False
Marker       :
PolicyGroups : {}
PolicyRoles  : {testRole}
PolicyUsers  : {Bob, Theresa}
```

- Pour plus de détails sur l'API, reportez-vous ListEntitiesForPolicyà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListGroupPolicies`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListGroupPolicies`.

CLI

AWS CLI

Pour répertorier toutes les politiques intégrées associées au groupe spécifié

La `list-group-policies` commande suivante répertorie les noms des politiques intégrées associées au groupe IAM nommé `Admins` dans le compte courant.

```
aws iam list-group-policies \
    --group-name Admins
```

Sortie :

```
{
    "PolicyNames": [
        "AdminRoot",
        "ExamplePolicy"
    ]
}
```

Pour plus d'informations, consultez Gestion des politiques IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir ListGroupPolitiques dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie une liste des politiques intégrées intégrées au groupe**`Testers`**. Pour obtenir les politiques gérées associées au groupe, utilisez la commande**`Get-IAMAttachedGroupPolicyList`**.

```
Get-IAMGroupPolicyList -GroupName Testers
```

Sortie :

```
Deny-Assume-S3-Role-In-Production
PowerUserAccess-Testers
```

- Pour plus de détails sur l'API, voir ListGroupPolitiques dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListGroups`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListGroups`.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour répertorier les groupes IAM du compte actuel

La commande `list-groups` suivante répertorie les groupes IAM du compte actuel.

```
aws iam list-groups
```

Sortie :

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-06-04T20:27:27.972Z",
            "GroupId": "AIDACKCEVSQ6C2EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "GroupName": "Admins"
        },
        {
            "Path": "/",
            "CreateDate": "2013-04-16T20:30:42Z",
            "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/S3-Admins",
            "GroupName": "S3-Admins"
        }
    ]
}
```

Pour plus d'informations, consultez la section Gestion des groupes IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// GroupWrapper encapsulates AWS Identity and Access Management (IAM) group
 actions
// used in the examples.
// It contains an IAM service client that is used to perform group actions.
type GroupWrapper struct {
 IamClient *iam.Client
}



// ListGroups lists up to maxGroups number of groups.
func (wrapper GroupWrapper) ListGroups(maxGroups int32) ([]types.Group, error) {
 var groups []types.Group
 result, err := wrapper.IamClient.ListGroups(context.TODO(),
 &iam.ListGroupsInput{
  MaxItems: aws.Int32(maxGroups),
 })
 if err != nil {
  log.Printf("Couldn't list groups. Here's why: %v\n", err)
 } else {
  groups = result.Groups
 }
 return groups, err
}
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des AWS SDK for Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les groupes.

```
import { ListGroupsCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listGroups() {
  const command = new ListGroupsCommand({
    MaxItems: 10,
  });

  let response = await client.send(command);

  while (response.Groups?.length) {
    for (const group of response.Groups) {
      yield group;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListGroupsCommand({
          Marker: response.Marker,
          MaxItems: 10,
        }),
      );
    } else {
      break;
```

```
    }
  }
}
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des AWS SDK for JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listGroups($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listGroupsArguments = [];
        if ($pathPrefix) {
            $listGroupsArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listGroupsArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listGroupsArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listGroups($listGroupsArguments);
    }
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des AWS SDK for PHP API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie une collection de tous les groupes IAM définis dans le fichier actuel Compte AWS.

```
Get-IAMGroupList
```

Sortie :

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE2
GroupName  : Developers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE3
GroupName  : Testers
Path       : /
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_groups(count):
    """
    Lists the specified number of groups for the account.

    :param count: The number of groups to list.
    """
    try:
        for group in iam.groups.limit(count):
            logger.info("Group: %s", group.name)
    except ClientError:
        logger.exception("Couldn't list groups for the account.")
        raise
```

- Pour plus de détails sur l'API, consultez ListGroupsle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# A class to manage IAM operations via the AWS SDK client
class IamGroupManager
  # Initializes the IamGroupManager class
  # @param iam_client [Aws::IAM::Client] An instance of the IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of groups for the account.
  # @param count [Integer] The maximum number of groups to list.
  # @return [Aws::IAM::Client::Response]
```

```
  def list_groups(count)
    response = @iam_client.list_groups(max_items: count)
    response.groups.each do |group|
      @logger.info("\t#{group.group_name}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list groups for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn list_groups(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListGroupsOutput, SdkError<ListGroupsError>> {
    let response = client
        .list_groups()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;
```

```
    Ok(response)
}
```

- Pour plus de détails sur l'API, voir ListGroupsla section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    public func listGroups() async throws -> [String] {
        var groupList: [String] = []
        var marker: String? = nil
        var isTruncated: Bool

        repeat {
            let input = ListGroupsInput(marker: marker)
            let output = try await client.listGroups(input: input)

            guard let groups = output.groups else {
                return groupList
            }

            for group in groups {
                if let name = group.groupName {
                    groupList.append(name)
                }
```

```
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        return groupList
    }
```

- Pour plus de détails sur l'API, reportez-vous ListGroupsà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListGroupsForUser`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListGroupsForUser`.

CLI

AWS CLI

Pour répertorier les groupes auxquels appartient un utilisateur IAM

La `list-groups-for-user` commande suivante affiche les groupes auxquels Bob appartient l'utilisateur IAM nommé.

```
aws iam list-groups-for-user \
    --user-name Bob
```

Sortie :

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:18:08Z",
            "GroupId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admin",
            "GroupName": "Admin"
        },
```

```
        {
            "Path": "/",
            "CreateDate": "2013-05-06T01:37:28Z",
            "GroupId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/s3-Users",
            "GroupName": "s3-Users"
        }
    ]
}
```

Pour plus d'informations, consultez la section Gestion des groupes IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir ListGroupsForUserla section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie la liste des groupes IAM auxquels **`David`** appartient l'utilisateur IAM.

```
Get-IAMGroupForUser -UserName David
```

Sortie :

```
Arn        : arn:aws:iam::123456789012:group/Administrators
CreateDate : 10/20/2014 10:06:24 AM
GroupId    : 6WCH4TRY3KIHIEXAMPLE1
GroupName  : Administrators
Path       : /

Arn        : arn:aws:iam::123456789012:group/Testers
CreateDate : 12/10/2014 3:39:11 PM
GroupId    : RHNZZGQJ7QHMAEXAMPLE2
GroupName  : Testers
Path       : /

Arn        : arn:aws:iam::123456789012:group/Developers
CreateDate : 12/10/2014 3:38:55 PM
GroupId    : ZU2EOWMK6WBZOEXAMPLE3
```

```
GroupName  : Developers
Path       : /
```

- Pour plus de détails sur l'API, consultez la section ListGroupsForUserRéférence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListInstanceProfiles`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListInstanceProfiles`.

CLI

AWS CLI

Pour répertorie les profils d'instance pour le compte

La `list-instance-profiles` commande suivante répertorie les profils d'instance associés au compte actuel.

```
aws iam list-instance-profiles
```

Sortie :

```
{
    "InstanceProfiles": [
        {
            "Path": "/",
            "InstanceProfileName": "example-dev-role",
            "InstanceProfileId": "AIPAIXEU4NUHUPEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-dev-role",
            "CreateDate": "2023-09-21T18:17:41+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-dev-role",
                    "RoleId": "AROAJ52OTH4H7LEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-dev-role",
                    "CreateDate": "2023-09-21T18:17:40+00:00",
```

```
                    "AssumeRolePolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
                }
            ]
        },
        {
            "Path": "/",
            "InstanceProfileName": "example-s3-role",
            "InstanceProfileId": "AIPAJVJVNRIQFREXAMPLE",
            "Arn": "arn:aws:iam::123456789012:instance-profile/example-s3-role",
            "CreateDate": "2023-09-21T18:18:50+00:00",
            "Roles": [
                {
                    "Path": "/",
                    "RoleName": "example-s3-role",
                    "RoleId": "AROAINUBC5O7XLEXAMPLE",
                    "Arn": "arn:aws:iam::123456789012:role/example-s3-role",
                    "CreateDate": "2023-09-21T18:18:49+00:00",
                    "AssumeRolePolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Principal": {
                                    "Service": "ec2.amazonaws.com"
                                },
                                "Action": "sts:AssumeRole"
                            }
                        ]
                    }
                }
            ]
        }
    ]
```

```
}
```

Pour de plus amples informations, consultez Utilisation de profils d'instance dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section ListInstanceProfils dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie une collection des profils d'instance définis dans le fichier actuel Compte AWS.

```
Get-IAMInstanceProfileList
```

Sortie :

```
Arn                 : arn:aws:iam::123456789012:instance-profile/ec2instancerole
CreateDate          : 2/17/2015 2:49:04 PM
InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
InstanceProfileName : ec2instancerole
Path                : /
Roles               : {ec2instancerole}
```

- Pour plus de détails sur l'API, consultez la section ListInstanceProfiles in AWS Tools for PowerShell Cmdlet Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListInstanceProfilesForRole`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListInstanceProfilesForRole`.

CLI

AWS CLI

Pour répertorier les profils d'instance pour un rôle IAM

La `list-instance-profiles-for-role` commande suivante répertorie les profils d'instance associés au rôle`Test-Role`.

```
aws iam list-instance-profiles-for-role \
    --role-name Test-Role
```

Sortie :

```
{
    "InstanceProfiles": [
        {
            "InstanceProfileId": "AIDGPMS9RO4H3FEXAMPLE",
            "Roles": [
                {
                    "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
                    "RoleId": "AIDACKCEVSQ6C2EXAMPLE",
                    "CreateDate": "2013-06-07T20:42:15Z",
                    "RoleName": "Test-Role",
                    "Path": "/",
                    "Arn": "arn:aws:iam::123456789012:role/Test-Role"
                }
            ],
            "CreateDate": "2013-06-07T21:05:24Z",
            "InstanceProfileName": "ExampleInstanceProfile",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:instance-profile/
ExampleInstanceProfile"
        }
    ]
}
```

Pour de plus amples informations, consultez Utilisation de profils d'instance dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section ListInstanceProfilesForRole in AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie les détails du profil d'instance associé au rôle**`ec2instancerole`**.

```
Get-IAMInstanceProfileForRole -RoleName ec2instancerole
```

Sortie :

```
      Arn                 : arn:aws:iam::123456789012:instance-profile/
ec2instancerole
      CreateDate          : 2/17/2015 2:49:04 PM
      InstanceProfileId   : HH36PTZQJUR32EXAMPLE1
      InstanceProfileName : ec2instancerole
      Path                : /
      Roles               : {ec2instancerole}
```

- Pour plus de détails sur l'API, consultez la section Référence des ListInstanceProfilesForrôles dans les AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListMfaDevices`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListMfaDevices`.

CLI

AWS CLI

Pour répertorier tous les appareils MFA d'un utilisateur spécifique

Cet exemple renvoie des informations sur le périphérique MFA attribué à l'utilisateur IAM. Bob

```
aws iam list-mfa-devices \
    --user-name Bob
```

Sortie :

```
{
    "MFADevices": [
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Bob",
            "EnableDate": "2019-10-28T20:37:09+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "GAKT12345678",
            "EnableDate": "2023-02-18T21:44:42+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey1-7XNL7NFNLZ123456789EXAMPLE",
            "EnableDate": "2023-09-19T02:25:35+00:00"
        },
        {
            "UserName": "Bob",
            "SerialNumber": "arn:aws:iam::123456789012:u2f/user/Bob/
fidosecuritykey2-VDRQTDBBN5123456789EXAMPLE",
            "EnableDate": "2023-09-19T01:49:18+00:00"
        }
    ]
}
```

Pour plus d'informations, consultez Utilisation de l'authentification multifacteur (MFA) dans AWS dans le AWS Guide de l'utilisateur IAM.

- Pour plus de détails sur l'API, consultez la section ListMfaDevices in AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie des informations sur le dispositif MFA attribué à l'utilisateur IAM. **`David`** Dans cet exemple, vous pouvez voir qu'il s'agit d'un périphérique virtuel car il **`SerialNumber`** s'agit d'un ARN au lieu du numéro de série réel d'un périphérique physique.

```
Get-IAMMFADevice -UserName David
```

Sortie :

```
EnableDate                  SerialNumber                            UserName
----------                  ------------                            --------
4/8/2015 9:41:10 AM         arn:aws:iam::123456789012:mfa/David     David
```

- Pour plus de détails sur l'API, consultez la section ListMfaDevices in AWS Tools for PowerShell Cmdlet Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListOpenIdConnectProviders`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListOpenIdConnectProviders`.

CLI

AWS CLI

Pour répertorier les informations relatives aux fournisseurs OpenID Connect dans le compte AWS

Cet exemple renvoie une liste des ARNS de tous les fournisseurs OpenID Connect définis dans AWS le compte courant.

```
aws iam list-open-id-connect-providers
```

Sortie :

```
{
    "OpenIDConnectProviderList": [
        {
            "Arn": "arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com"
        }
```

```
    ]
}
```

Pour plus d'informations, consultez la section Création de fournisseurs d'identité OpenID Connect (OIDC) dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section ListOpenIdConnectFournisseurs dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie une liste des ARNS de tous les fournisseurs OpenID Connect définis dans Compte AWS le présent document.

```
Get-IAMOpenIDConnectProviderList
```

Sortie :

```
Arn
---
arn:aws:iam::123456789012:oidc-provider/server.example.com
arn:aws:iam::123456789012:oidc-provider/another.provider.com
```

- Pour plus de détails sur l'API, consultez la section ListOpenIdConnectFournisseurs dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListPolicies`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListPolicies`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Gérer les politiques

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK for .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::listPolicies(const Aws::Client::ClientConfiguration
 &clientConfig) {
    const Aws::String DATE_FORMAT("%Y-%m-%d");
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListPoliciesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListPolicies(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam policies: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(64) << "Description" << std::setw(12) <<
                      "CreateDate" << std::endl;
            header = true;
        }

        const auto &policies = outcome.GetResult().GetPolicies();
        for (const auto &policy: policies) {
            std::cout << std::left << std::setw(55) <<
                      policy.GetPolicyName() << std::setw(30) <<
                      policy.GetPolicyId() << std::setw(80) << policy.GetArn() <<
                      std::setw(64) << policy.GetDescription() << std::setw(12)
 <<
                      policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
```

```
                        std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour répertorier les politiques gérées disponibles pour votre AWS compte

Cet exemple renvoie une collection des deux premières politiques gérées disponibles dans le AWS compte courant.

```
aws iam list-policies \
    --max-items 3
```

Sortie :

```
{
    "Policies": [
        {
            "PolicyName": "AWSCloudTrailAccessPolicy",
            "PolicyId": "ANPAXQE2B5PJ7YEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:policy/AWSCloudTrailAccessPolicy",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 0,
```

```
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2019-09-04T17:43:42+00:00",
            "UpdateDate": "2019-09-04T17:43:42+00:00"
        },
        {
            "PolicyName": "AdministratorAccess",
            "PolicyId": "ANPAIWMBCKSKIEE64ZLYK",
            "Arn": "arn:aws:iam::aws:policy/AdministratorAccess",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 6,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:46+00:00",
            "UpdateDate": "2015-02-06T18:39:46+00:00"
        },
        {
            "PolicyName": "PowerUserAccess",
            "PolicyId": "ANPAJYRXTHIB4FOVS3ZXS",
            "Arn": "arn:aws:iam::aws:policy/PowerUserAccess",
            "Path": "/",
            "DefaultVersionId": "v5",
            "AttachmentCount": 1,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:47+00:00",
            "UpdateDate": "2023-07-06T22:04:00+00:00"
        }
    ],
    "NextToken": "EXAMPLErZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiA4fQ=="
}
```


Pour de plus amples informations, veuillez consulter Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(maxPolicies int32) ([]types.Policy,
 error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(context.TODO(),
 &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK for Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les politiques.

```
import { ListPoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listPolicies() {
  const command = new ListPoliciesCommand({
    MaxItems: 10,
    OnlyAttached: false,
    // List only the customer managed policies in your Amazon Web Services
 account.
    Scope: "Local",
  });

  let response = await client.send(command);

  while (response.Policies?.length) {
    for (const policy of response.Policies) {
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListPoliciesCommand({
          Marker: response.Marker,
```

```
          MaxItems: 10,
          OnlyAttached: false,
          Scope: "Local",
        }),
      );
    } else {
      break;
    }
  }
}
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK for JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listPolicies($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listPoliciesArguments = [];
        if ($pathPrefix) {
            $listPoliciesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listPoliciesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listPoliciesArguments["MaxItems"] = $maxItems;
        }
```

```
        return $this->iamClient->listPolicies($listPoliciesArguments);
    }
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK for PHP API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie une collection des trois premières politiques gérées disponibles dans le AWS compte courant. Comme il n'**`-scope`**est pas spécifié, il utilise par défaut **`all`** et inclut à la fois des politiques AWS gérées et des politiques gérées par le client.

```
Get-IAMPolicyList -MaxItem 3
```

Sortie :

```
Arn              : arn:aws:iam::aws:policy/AWSDirectConnectReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:08 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : Z27SI6FQMGNQ2EXAMPLE1
PolicyName       : AWSDirectConnectReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:08 AM

Arn              : arn:aws:iam::aws:policy/AmazonGlacierReadOnlyAccess
AttachmentCount  : 0
CreateDate       : 2/6/2015 10:40:27 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : NJKMU274MET4EEXAMPLE2
PolicyName       : AmazonGlacierReadOnlyAccess
UpdateDate       : 2/6/2015 10:40:27 AM

Arn              : arn:aws:iam::aws:policy/AWSMarketplaceFullAccess
```

```
AttachmentCount  : 0
CreateDate       : 2/11/2015 9:21:45 AM
DefaultVersionId : v1
Description      :
IsAttachable     : True
Path             : /
PolicyId         : 5ULJSO2FYVPYGEXAMPLE3
PolicyName       : AWSMarketplaceFullAccess
UpdateDate       : 2/11/2015 9:21:45 AM
```

Exemple 2 : Cet exemple renvoie un ensemble des deux premières politiques gérées par le client disponibles dans le AWS compte courant. Il est utilisé **`-Scope local`** pour limiter la sortie aux seules politiques gérées par le client.

```
Get-IAMPolicyList -Scope local -MaxItem 2
```

Sortie :

```
Arn              : arn:aws:iam::123456789012:policy/MyLocalPolicy
AttachmentCount  : 0
CreateDate       : 2/12/2015 9:39:09 AM
DefaultVersionId : v2
Description      :
IsAttachable     : True
Path             : /
PolicyId         : SQVCBLC4VAOUCEXAMPLE4
PolicyName       : MyLocalPolicy
UpdateDate       : 2/12/2015 9:39:53 AM

Arn              : arn:aws:iam::123456789012:policy/policyforec2instancerole
AttachmentCount  : 1
CreateDate       : 2/17/2015 2:51:38 PM
DefaultVersionId : v11
Description      :
IsAttachable     : True
Path             : /
PolicyId         : X5JPBLJH2Z2SOEXAMPLE5
PolicyName       : policyforec2instancerole
UpdateDate       : 2/18/2015 8:52:31 AM
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
        return policies
```

- Pour plus de détails sur l'API, consultez ListPolicieslle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Cet exemple de module répertorie, crée, attache et détache les politiques de rôle.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
```

```
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
```

```
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn list_policies(
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
```

```
        .try_collect()
        .await?;

    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Pour plus de détails sur l'API, voir ListPoliciesla section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    public func listPolicies() async throws -> [MyPolicyRecord] {
        var policyList: [MyPolicyRecord] = []
        var marker: String? = nil
```

```
        var isTruncated: Bool

        repeat {
            let input = ListPoliciesInput(marker: marker)
            let output = try await client.listPolicies(input: input)

            guard let policies = output.policies else {
                return policyList
            }

            for policy in policies {
                guard   let name = policy.policyName,
                        let id = policy.policyId,
                        let arn = policy.arn else {
                    throw ServiceHandlerError.noSuchPolicy
                }
                policyList.append(MyPolicyRecord(name: name, id: id, arn: arn))
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        return policyList
    }
```

- Pour plus de détails sur l'API, reportez-vous ListPoliciesà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListPolicyVersions`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListPolicyVersions`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Gérer les politiques
- Restaurer une version de stratégie

CLI

AWS CLI

Pour répertorier les informations relatives aux versions de la politique gérée spécifiée

Cet exemple renvoie la liste des versions disponibles de la politique dont l'ARN est l'ARN`arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam list-policy-versions \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Sortie :

```
{
    "IsTruncated": false,
    "Versions": [
        {
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2015-06-02T23:19:44Z"
        },
        {
        "VersionId": "v1",
        "IsDefaultVersion": false,
        "CreateDate": "2015-06-02T22:30:47Z"
        }
    ]
}
```

Pour de plus amples informations, veuillez consulter Politiques et autorisations dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir ListPolicyVersions dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie la liste des versions disponibles de la politique dont l'ARN est l'ARN**`arn:aws:iam::123456789012:policy/MyManagedPolicy`**. Pour obtenir le

document de politique pour une version spécifique, utilisez la **`Get-IAMPolicyVersion`** commande et spécifiez **`VersionId`** celle que vous souhaitez.

```
Get-IAMPolicyVersionList -PolicyArn arn:aws:iam::123456789012:policy/
MyManagedPolicy
```

Sortie :

```
CreateDate                    Document                IsDefaultVersion
       VersionId
----------                    --------                ---------------
       ---------
2/12/2015 9:39:53 AM                                  True
       v2
2/12/2015 9:39:09 AM                                  False
       v1
```

- Pour plus de détails sur l'API, consultez la section ListPolicyVersions dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListRolePolicies`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListRolePolicies`.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
```

```
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }
```

- Pour plus de détails sur l'API, consultez la section ListRolePolitiques du manuel de référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour répertorier les politiques attachées à un rôle IAM

La commande `list-role-policies` suivante répertorie les noms des politiques des autorisations pour le rôle IAM spécifié.

```
aws iam list-role-policies \
    --role-name Test-Role
```

Sortie :

```
{
    "PolicyNames": [
        "ExamplePolicy"
```

```
    ]
}
```

Pour voir la politique d'approbation attachée à un rôle, utilisez la commande `get-role`. Pour voir les détails d'une politique des autorisations, utilisez la commande `get-role-policy`.

Pour plus d'informations, consultez Création de rôles IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir ListRolePolitiques dans AWS CLI Command Reference.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(roleName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
```

```
  policies = result.PolicyNames
 }
 return policies, err
}
```

- Pour plus de détails sur l'API, consultez la section ListRolePolitiques du manuel de référence des AWS SDK for Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les politiques.

```
import { ListRolePoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} roleName
 */
export async function* listRolePolicies(roleName) {
  const command = new ListRolePoliciesCommand({
    RoleName: roleName,
    MaxItems: 10,
  });

  let response = await client.send(command);
```

```
  while (response.PolicyNames?.length) {
    for (const policyName of response.PolicyNames) {
      yield policyName;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolePoliciesCommand({
          RoleName: roleName,
          MaxItems: 10,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Pour plus de détails sur l'API, consultez la section ListRolePolitiques du manuel de référence des AWS SDK for JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listRolePolicies($roleName, $marker = "", $maxItems = 0)
    {
        $listRolePoliciesArguments = ['RoleName' => $roleName];
        if ($marker) {
            $listRolePoliciesArguments['Marker'] = $marker;
```

```
        }
        if ($maxItems) {
            $listRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->customWaiter(function () use ($listRolePoliciesArguments) {
            return $this->iamClient-
>listRolePolicies($listRolePoliciesArguments);
        });
    }
```

- Pour plus de détails sur l'API, consultez la section ListRolePolitiques du manuel de référence des AWS SDK for PHP API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renvoie la liste des noms des politiques intégrées intégrées au rôle IAM. **`lamda_exec_role`** Pour voir les détails d'une politique intégrée, utilisez la commande**`Get-IAMRolePolicy`**.

```
Get-IAMRolePolicyList -RoleName lambda_exec_role
```

Sortie :

```
oneClick_lambda_exec_role_policy
```

- Pour plus de détails sur l'API, consultez la section ListRolePolitiques de référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_policies(role_name):
    """
    Lists inline policies for a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.policies.all():
            logger.info("Got inline policy %s.", policy.name)
    except ClientError:
        logger.exception("Couldn't list inline policies for %s.", role_name)
        raise
```

- Pour plus de détails sur l'API, consultez le ListRoledocument de référence de l'API Policies in AWS SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# Lists policy ARNs attached to a role
#
# @param role_name [String] The name of the role
# @return [Array<String>] List of policy ARNs
def list_attached_policy_arns(role_name)
  response = @iam_client.list_attached_role_policies(role_name: role_name)
  response.attached_policies.map(&:policy_arn)
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing policies attached to role: #{e.message}")
  []
end
```

- Pour plus de détails sur l'API, consultez la section ListRolePolitiques du manuel de référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn list_role_policies(
    client: &iamClient,
    role_name: &str,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolePoliciesOutput, SdkError<ListRolePoliciesError>> {
    let response = client
        .list_role_policies()
        .role_name(role_name)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Pour plus de détails sur l'API, consultez la section ListRolePolitiques du AWS SDK pour la référence de l'API Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
public func listRolePolicies(role: String) async throws -> [String] {
    var policyList: [String] = []
    var marker: String? = nil
    var isTruncated: Bool

    repeat {
        let input = ListRolePoliciesInput(
            marker: marker,
            roleName: role
        )
        let output = try await client.listRolePolicies(input: input)

        guard let policies = output.policyNames else {
            return policyList
        }

        for policy in policies {
            policyList.append(policy)
        }
        marker = output.marker
        isTruncated = output.isTruncated
    } while isTruncated == true
    return policyList
}
```

- Pour plus de détails sur l'API, consultez la section ListRolePolitiques du AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListRoleTags`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListRoleTags`.

CLI

AWS CLI

Pour répertorier les balises associées à un rôle

La `list-role-tags` commande suivante permet de récupérer la liste des balises associées au rôle spécifié.

```
aws iam list-role-tags \
    --role-name production-role
```

Sortie :

```
{
    "Tags": [
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

Pour plus d'informations, consultez la section Marquage des ressources IAM dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section ListRoleBalises dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple récupère le tag associé au rôle.

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- Pour plus de détails sur l'API, consultez la section Référence des ListRolebalises dans les AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListRoles`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListRoles`.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
```

```
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour répertorier les rôles IAM du compte actuel

La commande `list-roles` suivante répertorie les rôles IAM du compte actuel.

```
aws iam list-roles
```

Sortie :

```
{
    "Roles": [
        {
            "Path": "/",
            "RoleName": "ExampleRole",
            "RoleId": "AROAJ52OTH4H7LEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/ExampleRole",
            "CreateDate": "2017-09-12T19:23:36+00:00",
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
```

```
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        },
        {
            "Path": "/example_path/",
            "RoleName": "ExampleRoleWithPath",
            "RoleId": "AROAI4QRP7UFT7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/example_path/
ExampleRoleWithPath",
            "CreateDate": "2023-09-21T20:29:38+00:00",
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        }
    ]
}
```

Pour plus d'informations, consultez Création de rôles IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(maxRoles int32) ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(context.TODO(),
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des AWS SDK for Go API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les rôles.

```
import { ListRolesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listRoles() {
  const command = new ListRolesCommand({
    MaxItems: 10,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListRolesCommandOutput | undefined}
   */
  let response = await client.send(command);

  while (response?.Roles?.length) {
    for (const role of response.Roles) {
      yield role;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolesCommand({
          Marker: response.Marker,
        }),
```

```
      );
    } else {
      break;
    }
  }
}
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des AWS SDK for JavaScript API.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    /**
     * @param string $pathPrefix
     * @param string $marker
     * @param int $maxItems
     * @return Result
     * $roles = $service->listRoles();
     */
    public function listRoles($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listRolesArguments = [];
        if ($pathPrefix) {
            $listRolesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listRolesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
```

```
            $listRolesArguments["MaxItems"] = $maxItems;
        }
        return $this->iamClient->listRoles($listRolesArguments);
    }
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des AWS SDK for PHP API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple extrait une liste de tous les rôles IAM dans le. Compte AWS

```
Get-IAMRoleList
```

Exemple 2 : Cet extrait de code extrait une liste des rôles IAM du AWS compte, les affiche trois par trois, puis attend que vous appuyiez sur Entrée entre chaque groupe. Il transmet la **`Marker`** valeur de l'appel précédent pour indiquer où le groupe suivant doit commencer.

```
$nextMarker = $null
Do
{
    $results = Get-IAMRoleList -MaxItem 3 -Marker $nextMarker
    $nextMarker = $AWSHistory.LastServiceResponse.Marker
    $results
    Read-Host
} while ($nextMarker -ne $null)
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_roles(count):
    """
    Lists the specified number of roles for the account.

    :param count: The number of roles to list.
    """
    try:
        roles = list(iam.roles.limit(count=count))
        for role in roles:
            logger.info("Role: %s", role.name)
    except ClientError:
        logger.exception("Couldn't list roles for the account.")
        raise
    else:
        return roles
```

- Pour plus de détails sur l'API, consultez ListRolesle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
  # Lists IAM roles up to a specified count.
  # @param count [Integer] the maximum number of roles to list.
  # @return [Array<String>] the names of the roles.
  def list_roles(count)
    role_names = []
    roles_counted = 0

    @iam_client.list_roles.each_page do |page|
      page.roles.each do |role|
        break if roles_counted >= count
        @logger.info("\t#{roles_counted + 1}: #{role.role_name}")
        role_names << role.role_name
        roles_counted += 1
      end
      break if roles_counted >= count
    end

    role_names
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't list roles for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn list_roles(
    client: &iamClient,
    path_prefix: Option<String>,
```

```
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolesOutput, SdkError<ListRolesError>> {
    let response = client
        .list_roles()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;
    Ok(response)
}
```

- Pour plus de détails sur l'API, voir ListRolesla section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    public func listRoles() async throws -> [String] {
        var roleList: [String] = []
        var marker: String? = nil
        var isTruncated: Bool

        repeat {
            let input = ListRolesInput(marker: marker)
```

```
            let output = try await client.listRoles(input: input)

            guard let roles = output.roles else {
                return roleList
            }

            for role in roles {
                if let name = role.roleName {
                    roleList.append(name)
                }
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        return roleList
    }
```

- Pour plus de détails sur l'API, reportez-vous ListRolesà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListSAMLProviders`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListSAMLProviders`.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// List SAML authentication providers.
```

```
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }
```

- Pour les détails de l'API, consultez ListSAMLProviders dans la Référence de l'API AWS SDK for .NET .

CLI

AWS CLI

Pour répertorier les fournisseurs SAML dans le compte AWS

Cet exemple extrait la liste des fournisseurs SAML 2.0 créés dans le compte courant AWS .

```
aws iam list-saml-providers
```

Sortie :

```
{
    "SAMLProviderList": [
        {
            "Arn": "arn:aws:iam::123456789012:saml-provider/SAML-ADFS",
            "ValidUntil": "2015-06-05T22:45:14Z",
            "CreateDate": "2015-06-05T22:45:14Z"
        }
    ]
}
```

Pour plus d’informations, consultez Création de fournisseurs d’identité SAML IAM dans le Guide de l’utilisateur AWS IAM.

- Pour plus d’informations sur l’API, consultez ListSAMLProviders dans la Référence des commandes AWS CLI .

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders() ([]types.SAMLProviderListEntry,
 error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

- Pour les détails de l'API, consultez ListSAMLProviders dans la Référence de l'API AWS SDK for Go .

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les fournisseurs SAML.

```
import { ListSAMLProvidersCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listSamlProviders = async () => {
  const command = new ListSAMLProvidersCommand({});

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Pour les détails de l'API, consultez ListSAMLProviders dans la Référence de l'API AWS SDK for JavaScript .

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
```

```
$service = new IAMService();

    public function listSAMLProviders()
    {
        return $this->iamClient->listSAMLProviders();
    }
```

- Pour les détails de l'API, consultez ListSAMLProviders dans la Référence de l'API AWS SDK for PHP .

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple récupère la liste des fournisseurs SAML 2.0 créés dans le fichier actuel. Compte AWS Il renvoie l'ARN, la date de création et la date d'expiration pour chaque fournisseur SAML.

```
Get-IAMSAMLProviderList
```

Sortie :

```
Arn                                                 CreateDate
   ValidUntil
---                                                 ----------
   ----------
arn:aws:iam::123456789012:saml-provider/SAMLADFS    12/23/2014 12:16:55 PM
   12/23/2114 12:16:54 PM
```

- Pour plus de détails sur l'API, consultez ListSamlProviders dans AWS Tools for PowerShell la référence des applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_saml_providers(count):
    """
    Lists the SAML providers for the account.

    :param count: The maximum number of providers to list.
    """
    try:
        found = 0
        for provider in iam.saml_providers.limit(count):
            logger.info("Got SAML provider %s.", provider.arn)
            found += 1
        if found == 0:
            logger.info("Your account has no SAML providers.")
    except ClientError:
        logger.exception("Couldn't list SAML providers.")
        raise
```

- Pour les détails de l'API, consultez ListSAMLProviders dans la Référence de l'API du kit SDK AWS pour Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
class SamlProviderLister
  # Initializes the SamlProviderLister with IAM client and a logger.
  # @param iam_client [Aws::IAM::Client] The IAM client object.
  # @param logger [Logger] The logger object for logging output.
  def initialize(iam_client, logger = Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of SAML providers for the account.
  # @param count [Integer] The maximum number of providers to list.
  # @return [Aws::IAM::Client::Response]
  def list_saml_providers(count)
    response = @iam_client.list_saml_providers
    response.saml_provider_list.take(count).each do |provider|
      @logger.info("\t#{provider.arn}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list SAML providers. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Pour les détails de l'API, consultez ListSAMLProviders dans la Référence de l'API AWS SDK for Ruby .

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn list_saml_providers(
    client: &Client,
) -> Result<ListSamlProvidersOutput, SdkError<ListSAMLProvidersError>> {
    let response = client.list_saml_providers().send().await?;

    Ok(response)
}
```

- Pour les détails de l'API, consultez ListSAMLProviders dans la Référence de l'API du kit SDK AWS pour Rust.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListServerCertificates`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListServerCertificates`.

C++

Kit de développement logiciel (SDK) for C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::listServerCertificates(
        const Aws::Client::ClientConfiguration &clientConfig) {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";

    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListServerCertificatesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListServerCertificates(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list server certificates: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(14) << "UploadDate" << std::setw(14) <<
                      "ExpirationDate" << std::endl;
            header = true;
        }

        const auto &certificates =
                outcome.GetResult().GetServerCertificateMetadataList();

        for (const auto &certificate: certificates) {
            std::cout << std::left << std::setw(55) <<
                      certificate.GetServerCertificateName() << std::setw(30) <<
                      certificate.GetServerCertificateId() << std::setw(80) <<
                      certificate.GetArn() << std::setw(14) <<

 certificate.GetUploadDate().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::setw(14) <<

 certificate.GetExpiration().ToGmtString(DATE_FORMAT.c_str()) <<
                      std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
```

```
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Pour plus de détails sur l'API, consultez la section ListServerCertificats dans le AWS SDK for C++ manuel de référence des API.

CLI

AWS CLI

Pour répertorier les certificats de serveur de votre AWS compte

La `list-server-certificates` commande suivante répertorie tous les certificats de serveur stockés et disponibles pour utilisation dans votre AWS compte.

```
aws iam list-server-certificates
```

Sortie :

```
{
    "ServerCertificateMetadataList": [
        {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        {
            "Path": "/cloudfront/",
            "ServerCertificateName": "MyTestCert",
            "ServerCertificateId": "ASCAEXAMPLE456EXAMPLE",
```

```
            "Arn": "arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyTestCert",
            "UploadDate": "2015-04-21T18:14:16+00:00",
            "Expiration": "2018-01-14T17:52:36+00:00"
        }
    ]
}
```

Pour de plus amples informations, veuillez consulter Gestion des certificats de serveur dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section ListServerCertificats dans AWS CLI Command Reference.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les certificats.

```
import { ListServerCertificatesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listServerCertificates() {
  const command = new ListServerCertificatesCommand({});
  let response = await client.send(command);

  while (response.ServerCertificateMetadataList?.length) {
```

```
    for await (const cert of response.ServerCertificateMetadataList) {
      yield cert;
    }

    if (response.IsTruncated) {
      response = await client.send(new ListServerCertificatesCommand({}));
    } else {
      break;
    }
  }
}
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez la section ListServerCertificats dans le AWS SDK for JavaScript manuel de référence des API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listServerCertificates({}, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez la section ListServerCertificats dans le AWS SDK for JavaScript manuel de référence des API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple extrait la liste des certificats de serveur qui ont été téléchargés vers le serveur actuel Compte AWS.

```
Get-IAMServerCertificateList
```

Sortie :

```
Arn                   : arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyServerCertificate
Expiration            : 1/14/2018 9:52:36 AM
Path                  : /Org1/Org2/
ServerCertificateId   : ASCAJIFEXAMPLE17HQZYW
ServerCertificateName : MyServerCertificate
UploadDate            : 4/21/2015 11:14:16 AM
```

- Pour plus de détails sur l'API, consultez la section ListServerCertificats in AWS Tools for PowerShell Cmdlet Reference.

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez, mettez à jour et supprimez les certificats de serveur.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "ServerCertificateManager"
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key,
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info("No server certificates found.")
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
 #{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
```

```
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, consultez la section ListServerCertificats dans le AWS SDK for Ruby manuel de référence des API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListSigningCertificates`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListSigningCertificates`.

CLI

AWS CLI

Pour répertorier les certificats de signature d'un utilisateur IAM

La `list-signing-certificates` commande suivante répertorie les certificats de signature de l'utilisateur IAM nomméBob.

```
aws iam list-signing-certificates \
    --user-name Bob
```

Sortie :

```
{
    "Certificates": [
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-
body>-----END CERTIFICATE-----",
            "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
            "UploadDate": "2013-06-06T21:40:08Z"
        }
    ]
}
```

Pour plus d'informations, consultez la section Gérer les certificats de signature dans le guide de l'utilisateur Amazon EC2.

- Pour plus de détails sur l'API, consultez la section ListSigningCertificats dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple permet de récupérer des informations sur le certificat de signature associé à l'utilisateur nommé**Bob**.

```
Get-IAMSigningCertificate -UserName Bob
```

Sortie :

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
```

```
 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE

 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Pour plus de détails sur l'API, consultez la section ListSigningCertificats in AWS Tools for PowerShell Cmdlet Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListUserPolicies`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListUserPolicies`.

CLI

AWS CLI

Pour répertorier les politiques d'un utilisateur IAM

La commande `list-user-policies` suivante répertorie les politiques attachées à l'utilisateur IAM nommé Bob.

```
aws iam list-user-policies \
    --user-name Bob
```

Sortie :

```
{
    "PolicyNames": [
        "ExamplePolicy",
        "TestPolicy"
    ]
}
```

Pour plus d'informations, consultez la section Création d'un utilisateur IAM dans votre AWS compte dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir ListUserPolitiques dans AWS CLI Command Reference.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
```

```
 IamClient *iam.Client
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(userName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

- Pour plus de détails sur l'API, consultez la section ListUserPolitiques du manuel de référence des AWS SDK for Go API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple extrait la liste des noms des politiques intégrées intégrées au nom de l'utilisateur IAM. **`David`**

```
Get-IAMUserPolicyList -UserName David
```

Sortie :

```
Davids_IAM_Admin_Policy
```

- Pour plus de détails sur l'API, consultez la section ListUserPolitiques de référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListUserTags`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListUserTags`.

CLI

AWS CLI

Pour répertorier les tags attachés à un utilisateur

La `list-user-tags` commande suivante permet de récupérer la liste des balises associées à l'utilisateur IAM spécifié.

```
aws iam list-user-tags \
    --user-name alice
```

Sortie :

```
{
    "Tags": [
        {
            "Key": "Department",
            "Value": "Accounting"
        },
        {
            "Key": "DeptID",
            "Value": "12345"
        }
    ],
    "IsTruncated": false
}
```

Pour plus d'informations, consultez la section Marquage des ressources IAM dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section ListUserBalises dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple extrait le tag associé à l'utilisateur.

```
Get-IAMUserTagList -UserName joe
```

- Pour plus de détails sur l'API, consultez la section Référence des ListUserbalises dans les AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListUsers`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListUsers`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Créer des utilisateurs en lecture seule et en lecture-écriture

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
```

```
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
 ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK for .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_list_users
#
# List the IAM users in the account.
```

```
#
# Returns:
#       The list of users names
#    And:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_list_users() {
  local option OPTARG # Required to use getopts command in a function.
  local error_code
  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_users"
    echo "Lists the AWS Identity and Access Management (IAM) user in the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local response

  response=$(aws iam list-users \
    --output text \
    --query "Users[].UserName")
  error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-users operation failed.$response"
```

```
    return 1
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::listUsers(const Aws::Client::ClientConfiguration &clientConfig)
 {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListUsersRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListUsers(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam users:" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(32) << "Name" <<
                      std::setw(30) << "ID" << std::setw(64) << "Arn" <<
```

```
                    std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &users = outcome.GetResult().GetUsers();
        for (const auto &user: users) {
            std::cout << std::left << std::setw(32) << user.GetUserName() <<
                      std::setw(30) << user.GetUserId() << std::setw(64) <<
                      user.GetArn() << std::setw(20) <<
                      user.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
                      << std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour répertorier les utilisateurs IAM

La commande `list-users` suivante répertorie les utilisateurs IAM du compte actuel.

```
aws iam list-users
```

Sortie :

```
{
    "Users": [
```

```
        {
            "UserName": "Adele",
            "Path": "/",
            "CreateDate": "2013-03-07T05:14:48Z",
            "UserId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Adele"
        },
        {
            "UserName": "Bob",
            "Path": "/",
            "CreateDate": "2012-09-21T23:03:13Z",
            "UserId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Bob"
        }
    ]
}
```

Pour plus d'informations, consultez la section Liste des utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS CLI commandes.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}
```

```
// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(maxUsers int32) ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(context.TODO(), &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK for Go API.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.AttachedPermissionsBoundary;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListUsersRequest;
import software.amazon.awssdk.services.iam.model.ListUsersResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.User;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
```

```
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListUsers {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listAllUsers(iam);
        System.out.println("Done");
        iam.close();
    }

    public static void listAllUsers(IamClient iam) {
        try {
            boolean done = false;
            String newMarker = null;
            while (!done) {
                ListUsersResponse response;
                if (newMarker == null) {
                    ListUsersRequest request =
 ListUsersRequest.builder().build();
                    response = iam.listUsers(request);
                } else {
                    ListUsersRequest request = ListUsersRequest.builder()
                            .marker(newMarker)
                            .build();

                    response = iam.listUsers(request);
                }

                for (User user : response.users()) {
                    System.out.format("\n Retrieved user %s", user.userName());
                    AttachedPermissionsBoundary permissionsBoundary =
 user.permissionsBoundary();
                    if (permissionsBoundary != null)
                        System.out.format("\n Permissions boundary details %s",

 permissionsBoundary.permissionsBoundaryTypeAsString());
                }
```

```
                if (!response.isTruncated()) {
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez les utilisateurs.

```
import { ListUsersCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listUsers = async () => {
  const command = new ListUsersCommand({ MaxItems: 10 });

  const response = await client.send(command);
  response.Users?.forEach(({ UserName, CreateDate }) => {
    console.log(`${UserName} created on: ${CreateDate}`);
```

```
  });
  return response;
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 10,
};

iam.listUsers(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var users = data.Users || [];
    users.forEach(function (user) {
      console.log("User " + user.UserName + " created", user.CreateDate);
    });
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK for JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun listAllUsers() {
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listUsers(ListUsersRequest { })
        response.users?.forEach { user ->
            println("Retrieved user ${user.userName}")
            val permissionsBoundary = user.permissionsBoundary
            if (permissionsBoundary != null) {
                println("Permissions boundary details
 ${permissionsBoundary.permissionsBoundaryType}")
            }
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section AWS SDK pour la référence de l'API Kotlin.

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listUsers($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listUsersArguments = [];
        if ($pathPrefix) {
            $listUsersArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listUsersArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listUsersArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listUsers($listUsersArguments);
    }
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK for PHP API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple récupère une collection d'utilisateurs dans le fichier actuel Compte AWS.

```
Get-IAMUserList
```

Sortie :

```
      Arn              : arn:aws:iam::123456789012:user/Administrator
      CreateDate       : 10/16/2014 9:03:09 AM
      PasswordLastUsed : 3/4/2015 12:12:33 PM
      Path             : /
      UserId           : 7K3GJEANSKZF2EXAMPLE1
      UserName         : Administrator

      Arn              : arn:aws:iam::123456789012:user/Bob
      CreateDate       : 4/6/2015 12:54:42 PM
      PasswordLastUsed : 1/1/0001 12:00:00 AM
      Path             : /
      UserId           : L3EWNONDOM3YUEXAMPLE2
      UserName         : bab

      Arn              : arn:aws:iam::123456789012:user/David
      CreateDate       : 12/10/2014 3:39:27 PM
      PasswordLastUsed : 3/19/2015 8:44:04 AM
      Path             : /
      UserId           : Y4FKWQCXTA52QEXAMPLE3
      UserName         : David
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def list_users():
    """
```

```
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users
```

- Pour plus de détails sur l'API, consultez ListUsersle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# Lists all users in the AWS account
#
# @return [Array<Aws::IAM::Types::User>] An array of user objects
def list_users
  users = []
  @iam_client.list_users.each_page do |page|
    page.users.each do |user|
      users << user
    end
  end
  users
rescue Aws::IAM::Errors::ServiceError => e
```

```
    @logger.error("Error listing users: #{e.message}")
    []
  end
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
pub async fn list_users(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListUsersOutput, SdkError<ListUsersError>> {
    let response = client
        .list_users()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;
    Ok(response)
}
```

- Pour plus de détails sur l'API, voir ListUsersla section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
public func listUsers() async throws -> [MyUserRecord] {
    var userList: [MyUserRecord] = []
    var marker: String? = nil
    var isTruncated: Bool

    repeat {
        let input = ListUsersInput(marker: marker)
        let output = try await client.listUsers(input: input)

        guard let users = output.users else {
            return userList
        }

        for user in users {
            if let id = user.userId, let name = user.userName {
                userList.append(MyUserRecord(id: id, name: name))
            }
        }
        marker = output.marker
        isTruncated = output.isTruncated
    } while isTruncated == true
    return userList
}
```

- Pour plus de détails sur l'API, reportez-vous ListUsersà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`ListVirtualMfaDevices`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ListVirtualMfaDevices`.

CLI

AWS CLI

Pour répertorier les appareils MFA virtuels

La `list-virtual-mfa-devices` commande suivante répertorie les périphériques MFA virtuels qui ont été configurés pour le compte actuel.

```
aws iam list-virtual-mfa-devices
```

Sortie :

```
{
    "VirtualMFADevices": [
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/ExampleMFADevice"
        },
        {
            "SerialNumber": "arn:aws:iam::123456789012:mfa/Fred"
        }
    ]
}
```

Pour plus d'informations, consultez la section Activation d'un dispositif d'authentification multifactorielle virtuelle (MFA) dans AWS le guide de l'utilisateur IAM.

- Pour plus de détails sur l'API, reportez-vous ListVirtualMfaDevicesà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple extrait un ensemble de dispositifs MFA virtuels assignés aux utilisateurs du compte. AWS La **`User`** propriété de chacun est un objet contenant les détails de l'utilisateur IAM auquel l'appareil est attribué.

```
Get-IAMVirtualMFADevice -AssignmentStatus Assigned
```

Sortie :

```
Base32StringSeed :
EnableDate       : 4/13/2015 12:03:42 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/David
User             : Amazon.IdentityManagement.Model.User

Base32StringSeed :
EnableDate       : 4/13/2015 12:06:41 PM
QRCodePNG        :
SerialNumber     : arn:aws:iam::123456789012:mfa/root-account-mfa-device
User             : Amazon.IdentityManagement.Model.User
```

- Pour plus de détails sur l'API, reportez-vous ListVirtualMfaDevicesà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`PutGroupPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`PutGroupPolicy`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Créer un groupe et ajouter un utilisateur

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, consultez PutGroupla section Politique dans le Guide de référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour ajouter une politique à un groupe

La commande `put-group-policy` suivante ajoute une politique au groupe IAM nommé `Admins`.

```
aws iam put-group-policy \
    --group-name Admins \
    --policy-document file://AdminPolicy.json \
    --policy-name AdminRoot
```

Cette commande ne produit aucun résultat.

La politique est définie sous la forme d'un document JSON dans le fichier AdminPolicy.json. (Le nom et l'extension du fichier n'ont aucune importance.)

Pour plus d'informations, consultez Gestion des politiques IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez PutGroupla section Politique dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée une politique intégrée nommée **`AppTesterPolicy`** et l'intègre dans le groupe IAM. **`AppTesters`** Si une politique intégrée portant le même nom existe déjà, elle est remplacée. Le contenu de la politique JSON est fourni dans le fichier**`apptesterpolicy.json`**. Notez que vous devez utiliser le **`-Raw`** paramètre pour traiter correctement le contenu du fichier JSON.

```
Write-IAMGroupPolicy -GroupName AppTesters -PolicyName AppTesterPolicy -
PolicyDocument (Get-Content -Raw apptesterpolicy.json)
```

- Pour plus de détails sur l'API, consultez PutGroupla section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **PutRolePermissionsBoundary** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliserPutRolePermissionsBoundary.

CLI

AWS CLI

Exemple 1 : pour appliquer une limite d'autorisation basée sur une politique personnalisée à un rôle IAM

L'put-role-permissions-boundaryexemple suivant applique la politique personnalisée intern-boundary nommée limite d'autorisations pour le rôle IAM spécifié.

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --role-name lambda-application-role
```

Cette commande ne produit aucun résultat.

Exemple 2 : pour appliquer une limite d'autorisation basée sur une politique AWS gérée à un rôle IAM

L'put-role-permissions-boundaryexemple suivant applique la PowerUserAccess politique AWS gérée comme limite d'autorisations pour le rôle IAM spécifié.

```
aws iam put-role-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --role-name x-account-admin
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez Modification d'un rôle dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous PutRolePermissionsBoundaryà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple montre comment définir la limite d'autorisation pour un rôle IAM. Vous pouvez définir des politiques AWS gérées ou des politiques personnalisées comme limite d'autorisation.

```
Set-IAMRolePermissionsBoundary -RoleName MyRoleName -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Pour plus de détails sur l'API, consultez la section PutRolePermissionsBoundaryRéférence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`PutRolePolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`PutRolePolicy`.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
```

```
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, consultez PutRolela section Politique dans le Guide de référence des AWS SDK for .NET API.

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::putRolePolicy(
        const Aws::String &roleName,
        const Aws::String &policyName,
        const Aws::String &policyDocument,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient iamClient(clientConfig);
    Aws::IAM::Model::PutRolePolicyRequest request;

    request.SetRoleName(roleName);
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(policyDocument);
```

```
    Aws::IAM::Model::PutRolePolicyOutcome outcome =
 iamClient.PutRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error putting policy on role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully put the role policy." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, consultez PutRolela section Politique dans le Guide de référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour attacher une politique d’autorisation à un rôle IAM

La commande `put-role-policy` suivante ajoute une politique d’autorisation au rôle nommé `Test-Role`.

```
aws iam put-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

Cette commande ne produit aucun résultat.

La politique est définie sous la forme d'un document JSON dans le fichier AdminPolicy.json. (Le nom et l’extension du fichier n’ont aucune importance.)

Pour attacher une politique d’approbation à un rôle, utilisez la commande `update-assume-role-policy`.

Pour plus d’informations, consultez Modification d’un rôle dans le Guide de l’utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez PutRolela section Politique dans AWS CLI Command Reference.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { PutRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const examplePolicyDocument = JSON.stringify({
  Version: "2012-10-17",
  Statement: [
    {
      Sid: "VisualEditor0",
      Effect: "Allow",
      Action: [
        "s3:ListBucketMultipartUploads",
        "s3:ListBucketVersions",
        "s3:ListBucket",
        "s3:ListMultipartUploadParts",
      ],
      Resource: "arn:aws:s3:::some-test-bucket",
    },
    {
      Sid: "VisualEditor1",
      Effect: "Allow",
      Action: [
        "s3:ListStorageLensConfigurations",
        "s3:ListAccessPointsForObjectLambda",
        "s3:ListAllMyBuckets",
        "s3:ListAccessPoints",
        "s3:ListJobs",
        "s3:ListMultiRegionAccessPoints",
      ],
      Resource: "*",
```

```
    },
  ],
});

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 * @param {string} policyDocument
 */
export const putRolePolicy = async (roleName, policyName, policyDocument) => {
  const command = new PutRolePolicyCommand({
    RoleName: roleName,
    PolicyName: policyName,
    PolicyDocument: policyDocument,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Pour plus de détails sur l'API, consultez PutRolela section Politique dans le Guide de référence des AWS SDK for JavaScript API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée une politique intégrée nommée **FedTesterRolePolicy** et l'intègre dans le rôle IAM. **FedTesterRole** Si une politique intégrée portant le même nom existe déjà, elle est remplacée. Le contenu de la politique JSON provient du fichier**FedTesterPolicy.json**. Notez que vous devez utiliser le **-Raw** paramètre pour traiter correctement le contenu du fichier JSON.

```
Write-IAMRolePolicy -RoleName FedTesterRole -PolicyName FedTesterRolePolicy -
PolicyDocument (Get-Content -Raw FedTesterPolicy.json)
```

- Pour plus de détails sur l'API, consultez PutRolela section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`PutUserPermissionsBoundary`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`PutUserPermissionsBoundary`.

CLI

AWS CLI

Exemple 1 : pour appliquer une limite d'autorisation basée sur une politique personnalisée à un utilisateur IAM

L'`put-user-permissions-boundary`exemple suivant applique une politique personnalisée `intern-boundary` nommée limite d'autorisations pour l'utilisateur IAM spécifié.

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::123456789012:policy/intern-boundary \
    --user-name intern
```

Cette commande ne produit aucun résultat.

Exemple 2 : pour appliquer une limite d'autorisation basée sur une politique AWS gérée à un utilisateur IAM

L'`put-user-permissions-boundary`exemple suivant applique la politique AWS gérée `PowerUserAccess` nommée limite d'autorisations pour l'utilisateur IAM spécifié.

```
aws iam put-user-permissions-boundary \
    --permissions-boundary arn:aws:iam::aws:policy/PowerUserAccess \
    --user-name developer
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la rubrique Ajout et suppression d'autorisations basées sur l'identité IAM du Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous PutUserPermissionsBoundaryà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple montre comment définir la limite d'autorisation pour l'utilisateur. Vous pouvez définir des politiques AWS gérées ou des politiques personnalisées comme limite d'autorisation.

```
Set-IAMUserPermissionsBoundary -UserName joe -PermissionsBoundary
 arn:aws:iam::123456789012:policy/intern-boundary
```

- Pour plus de détails sur l'API, reportez-vous PutUserPermissionsBoundaryà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`PutUserPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`PutUserPolicy`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Créer un utilisateur et assumer d'un rôle

CLI

AWS CLI

Pour attacher une politique à un utilisateur IAM

La commande `put-user-policy` suivante attache une politique à l’utilisateur IAM nommé `Bob`.

```
aws iam put-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

Cette commande ne produit aucun résultat.

La politique est définie sous la forme d'un document JSON dans le fichier AdminPolicy.json. (Le nom et l'extension du fichier n'ont aucune importance.)

Pour plus d'informations, consultez la rubrique Ajout et suppression d'autorisations basées sur l'identité IAM du Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez PutUserla section Politique dans AWS CLI Command Reference.

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
```

```
func (wrapper UserWrapper) CreateUserPolicy(userName string, policyName string,
 actions []string,
  roleArn string) error {
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(context.TODO(),
 &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

- Pour plus de détails sur l'API, consultez PutUserla section Politique dans le Guide de référence des AWS SDK for Go API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple crée une politique intégrée nommée **`EC2AccessPolicy`** et l'intègre à l'utilisateur IAM. **`Bob`** Si une politique intégrée portant le même nom existe déjà, elle est remplacée. Le contenu de la politique JSON provient du fichier**`EC2AccessPolicy.json`**.

Notez que vous devez utiliser le **`-Raw`** paramètre pour traiter correctement le contenu du fichier JSON.

```
Write-IAMUserPolicy -UserName Bob -PolicyName EC2AccessPolicy -PolicyDocument
 (Get-Content -Raw EC2AccessPolicy.json)
```

- Pour plus de détails sur l'API, consultez PutUserla section Politique dans la référence des AWS Tools for PowerShell applets de commande.

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Creates an inline policy for a specified user.
 # @param username [String] The name of the IAM user.
 # @param policy_name [String] The name of the policy to create.
 # @param policy_document [String] The JSON policy document.
 # @return [Boolean]
 def create_user_policy(username, policy_name, policy_document)
   @iam_client.put_user_policy({
     user_name: username,
     policy_name: policy_name,
     policy_document: policy_document
   })
   @logger.info("Policy #{policy_name} created for user #{username}.")
   true
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Couldn't create policy #{policy_name} for user #{username}.
Here's why:")
   @logger.error("\t#{e.code}: #{e.message}")
   false
 end
```

- Pour plus de détails sur l'API, consultez PutUserla section Politique dans le Guide de référence des AWS SDK for Ruby API.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
   func putUserPolicy(policyDocument: String, policyName: String, user:
IAMClientTypes.User) async throws {
       let input = PutUserPolicyInput(
           policyDocument: policyDocument,
           policyName: policyName,
           userName: user.userName
       )
       do {
           _ = try await iamClient.putUserPolicy(input: input)
       } catch {
           throw error
       }
   }
```

- Pour plus de détails sur l'API, consultez PutUserla section Politique du AWS SDK pour la référence à l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`RemoveClientIdFromOpenIdConnectProvider`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`RemoveClientIdFromOpenIdConnectProvider`.

CLI

AWS CLI

Pour supprimer l'ID client spécifié de la liste des identifiants clients enregistrés pour le fournisseur IAM OpenID Connect spécifié

Cet exemple supprime l'ID client `My-TestApp-3` de la liste des ID client associés au fournisseur IAM OIDC dont l'ARN est l'ARN. `arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`

```
aws iam remove-client-id-from-open-id-connect-provider
    --client-id My-TestApp-3 \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Création de fournisseurs d'identité OpenID Connect (OIDC) dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous RemoveClientIdFromOpenIdConnectProviderà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime l'ID client **`My-TestApp-3`** de la liste des ID clients associés au fournisseur IAM OIDC dont l'ARN est l'ARN. **`arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`**

```
Remove-IAMClientIDFromOpenIDConnectProvider -ClientID My-TestApp-3
 -OpenIDConnectProviderArn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com
```

- Pour plus de détails sur l'API, reportez-vous RemoveClientIdFromOpenIdConnectProviderà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`RemoveRoleFromInstanceProfile`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`RemoveRoleFromInstanceProfile`.

CLI

AWS CLI

Pour supprimer un rôle d'un profil d'instance

La `remove-role-from-instance-profile` commande suivante supprime le rôle nommé `Test-Role` du profil d'instance nommé`ExampleInstanceProfile`.

```
aws iam remove-role-from-instance-profile \
    --instance-profile-name ExampleInstanceProfile \
    --role-name Test-Role
```

Pour de plus amples informations, consultez Utilisation de profils d'instance dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section RemoveRoleFromInstanceProfil dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime le rôle nommé dans le profil **`MyNewRole`** d'instance EC2 nommé. **`MyNewRole`** Un profil d'instance créé dans la console IAM porte toujours le même

nom que le rôle, comme dans cet exemple. Si vous les créez dans l'API ou la CLI, ils peuvent porter des noms différents.

```
Remove-IAMRoleFromInstanceProfile -InstanceProfileName MyNewRole -RoleName
 MyNewRole -Force
```

- Pour plus de détails sur l'API, consultez la section RemoveRoleFromInstanceProfil dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`RemoveUserFromGroup`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`RemoveUserFromGroup`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Créer un groupe et ajouter un utilisateur

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
```

```
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Pour plus de détails sur l'API, reportez-vous RemoveUserFromGroupà la section Référence des AWS SDK for .NET API.

CLI

AWS CLI

Pour supprimer un utilisateur d'un groupe IAM

La commande `remove-user-from-group` suivante supprime l'utilisateur nommé Bob du groupe IAM nommé `Admins`.

```
aws iam remove-user-from-group \
    --user-name Bob \
    --group-name Admins
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, veuillez consulter Ajout et suppression d'utilisateurs dans un groupe IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous RemoveUserFromGroupà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime l'utilisateur IAM **Bob** du groupe**Testers**.

```
Remove-IAMUserFromGroup -GroupName Testers -UserName Bob
```

Exemple 2 : Cet exemple permet de rechercher tous les groupes dont un utilisateur IAM **Theresa** est membre, puis **Theresa** de les supprimer de ces groupes.

```
$groups = Get-IAMGroupForUser -UserName Theresa
foreach ($group in $groups) { Remove-IAMUserFromGroup -GroupName $group.GroupName
 -UserName Theresa -Force }
```

Exemple 3 : Cet exemple montre une autre méthode pour supprimer l'utilisateur IAM **Bob** du **Testers** groupe.

```
Get-IAMGroupForUser -UserName Bob | Remove-IAMUserFromGroup -UserName Bob -
GroupName Testers -Force
```

- Pour plus de détails sur l'API, reportez-vous RemoveUserFromGroupà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **ResyncMfaDevice** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`ResyncMfaDevice`.

CLI

AWS CLI

Pour synchroniser un appareil MFA

L'`resync-mfa-device`exemple suivant synchronise le périphérique MFA associé à l'utilisateur IAM Bob et dont l'ARN `arn:aws:iam::123456789012:mfa/BobsMFADevice` est associé à un programme d'authentification qui a fourni les deux codes d'authentification.

```
aws iam resync-mfa-device \
    --user-name Bob \
    --serial-number arn:aws:iam::210987654321:mfa/BobsMFADevice \
    --authentication-code1 123456 \
    --authentication-code2 987654
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez Utilisation de l'authentification multifacteur (MFA) dans AWS dans le AWS Guide de l'utilisateur IAM.

- Pour plus de détails sur l'API, consultez ResyncMfaDevice in AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple synchronise le dispositif MFA associé à l'utilisateur IAM **Bob** et dont l'ARN **arn:aws:iam::123456789012:mfa/bob** est associé à un programme d'authentification qui a fourni les deux codes d'authentification.

```
Sync-IAMMFADevice -SerialNumber arn:aws:iam::123456789012:mfa/theresa -
AuthenticationCode1 123456 -AuthenticationCode2 987654 -UserName Bob
```

Exemple 2 : Cet exemple synchronise le dispositif MFA IAM associé à l'utilisateur IAM **Theresa** avec un périphérique physique qui possède le numéro de série **ABCD12345678** et qui a fourni les deux codes d'authentification.

```
Sync-IAMMFADevice -SerialNumber ABCD12345678 -AuthenticationCode1 123456 -
AuthenticationCode2 987654 -UserName Theresa
```

- Pour plus de détails sur l'API, voir ResyncMfaDevice in AWS Tools for PowerShell Cmdlet Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`SetDefaultPolicyVersion`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`SetDefaultPolicyVersion`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Gérer les politiques
- Restaurer une version de stratégie

CLI

AWS CLI

Pour définir la version spécifiée de la stratégie spécifiée comme version par défaut de la stratégie.

Cet exemple définit la `v2` version de la politique dont l'`arn:aws:iam::123456789012:policy/MyPolicy`ARN est la version active par défaut.

```
aws iam set-default-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Pour de plus amples informations, veuillez consulter Policies and permissions in IAM (Stratégies et autorisations dans IAM) dans le AWS IAM Guide de l'utilisateur.

- Pour plus de détails sur l'API, reportez-vous SetDefaultPolicyVersionà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple définit la **`v2`** version de la politique dont l'**`arn:aws:iam::123456789012:policy/MyPolicy`**ARN est la version active par défaut.

```
Set-IAMDefaultPolicyVersion -PolicyArn arn:aws:iam::123456789012:policy/MyPolicy
 -VersionId v2
```

- Pour plus de détails sur l'API, reportez-vous SetDefaultPolicyVersionà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`TagRole`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`TagRole`.

CLI

AWS CLI

Pour ajouter un tag à un rôle

La `tag-role` commande suivante ajoute une balise avec un nom de département au rôle spécifié.

```
aws iam tag-role --role-name my-role \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Marquage des ressources IAM dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous TagRoleà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : cet exemple ajoute une balise au rôle dans le service de gestion des identités

```
Add-IAMRoleTag -RoleName AdminRoleacess -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Pour plus de détails sur l'API, reportez-vous TagRoleà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`TagUser`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`TagUser`.

CLI

AWS CLI

Pour ajouter un tag à un utilisateur

La `tag-user` commande suivante ajoute une balise avec le département associé à l'utilisateur spécifié.

```
aws iam tag-user \
    --user-name alice \
    --tags '{"Key": "Department", "Value": "Accounting"}'
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Marquage des ressources IAM dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous TagUserà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : cet exemple ajoute une balise à l'utilisateur dans le service de gestion des identités

```
Add-IAMUserTag -UserName joe -Tag @{ Key = 'abac'; Value = 'testing'}
```

- Pour plus de détails sur l'API, reportez-vous TagUserà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UntagRole`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UntagRole`.

CLI

AWS CLI

Pour supprimer un tag d'un rôle

La `untag-role` commande suivante supprime toute balise portant le nom de clé « Department » du rôle spécifié.

```
aws iam untag-role \
    --role-name my-role \
    --tag-keys Department
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Marquage des ressources IAM dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UntagRoleà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime la balise du rôle nommé « MyRole Nom » avec la clé de balise « abac ». Pour supprimer plusieurs balises, fournissez une liste de clés de balises séparées par des virgules.

```
Remove-IAMRoleTag -RoleName MyRoleName -TagKey "abac","xyzw"
```

- Pour plus de détails sur l'API, consultez la section UntagRoleRéférence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UntagUser`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UntagUser`.

CLI

AWS CLI

Pour supprimer un tag d'un utilisateur

La `untag-user` commande suivante supprime toute balise portant le nom de clé « Department » de l'utilisateur spécifié.

```
aws iam untag-user \
    --user-name alice \
    --tag-keys Department
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Marquage des ressources IAM dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UntagUserà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple supprime le tag de l'utilisateur nommé « joe » avec les clés de tag « abac » et « xyzw ». Pour supprimer plusieurs balises, fournissez une liste de clés de balises séparées par des virgules.

```
Remove-IAMUserTag -UserName joe -TagKey "abac","xyzw"
```

- Pour plus de détails sur l'API, reportez-vous UntagUserà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UpdateAccessKey`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UpdateAccessKey`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Gérer des clés d'accès

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::updateAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
                                  Aws::IAM::Model::StatusType status,
                                  const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::UpdateAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);
    request.SetStatus(status);

    auto outcome = iam.UpdateAccessKey(request);
    if (outcome.IsSuccess()) {
        std::cout << "Successfully updated status of access key "
                  << accessKeyID << " for user " << userName << std::endl;
    }
```

```
    else {
        std::cerr << "Error updated status of access key " << accessKeyID <<
                  " for user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, voir UpdateAccessKey in AWS SDK for C++ API Reference.

CLI

AWS CLI

Pour activer ou désactiver une clé d’accès pour un utilisateur IAM

La commande `update-access-key` suivante désactive la clé d’accès spécifiée (un ID de clé d’accès rapide et une clé d’accès secrète) pour l’utilisateur IAM nommé Bob.

```
aws iam update-access-key \
    --access-key-id AKIAIOSFODNN7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

Cette commande ne produit aucun résultat.

La désactivation de la clé signifie qu'elle ne peut pas être utilisée pour un accès programmatique à. AWS Cependant, la clé est toujours disponible et peut être réactivée.

Pour de plus amples informations, veuillez consulter Gestion des clés d’accès pour les utilisateurs IAM dans le Guide de l’utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir UpdateAccessKey in AWS CLI Command Reference.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.StatusType;
import software.amazon.awssdk.services.iam.model.UpdateAccessKeyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateAccessKey {

    private static StatusType statusType;

    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username> <accessId> <status>\s

                Where:
                    username - The name of the user whose key you want to update.
\s
                    accessId - The access key ID of the secret access key you
 want to update.\s
                    status - The status you want to assign to the secret access
 key.\s
```

```
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        String accessId = args[1];
        String status = args[2];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateKey(iam, username, accessId, status);
        System.out.println("Done");
        iam.close();
    }

    public static void updateKey(IamClient iam, String username, String accessId,
String status) {
        try {
            if (status.toLowerCase().equalsIgnoreCase("active")) {
                statusType = StatusType.ACTIVE;
            } else if (status.toLowerCase().equalsIgnoreCase("inactive")) {
                statusType = StatusType.INACTIVE;
            } else {
                statusType = StatusType.UNKNOWN_TO_SDK_VERSION;
            }

            UpdateAccessKeyRequest request = UpdateAccessKeyRequest.builder()
                    .accessKeyId(accessId)
                    .userName(username)
                    .status(statusType)
                    .build();

            iam.updateAccessKey(request);
            System.out.printf("Successfully updated the status of access key %s
to" +
                    "status %s for user %s", accessId, status, username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, voir UpdateAccessKey in AWS SDK for Java 2.x API Reference.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Mettez à jour la clé d'accès.

```
import {
  UpdateAccessKeyCommand,
  IAMClient,
  StatusType,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const updateAccessKey = (userName, accessKeyId) => {
  const command = new UpdateAccessKeyCommand({
    AccessKeyId: accessKeyId,
    Status: StatusType.Inactive,
    UserName: userName,
  });

  return client.send(command);
```

```
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, voir UpdateAccessKey in AWS SDK for JavaScript API Reference.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
  Status: "Active",
  UserName: "USER_NAME",
};

iam.updateAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.

- Pour plus de détails sur l'API, voir UpdateAccessKey in AWS SDK for JavaScript API Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple modifie le statut de la clé d'accès **AKIAIOSFODNN7EXAMPLE** pour l'utilisateur IAM nommé **Bob** à**Inactive**.

```
Update-IAMAccessKey -UserName Bob -AccessKeyId AKIAIOSFODNN7EXAMPLE -Status
 Inactive
```

- Pour plus de détails sur l'API, consultez UpdateAccessla section Key in AWS Tools for PowerShell Cmdlet Reference.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
    :param activate: When True, the key is activated. Otherwise, the key is
 deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
```

```
        else:
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
key_id)
    except ClientError:
        logger.exception(
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
key_id
        )
        raise
```

- Pour plus de détails sur l'API, consultez le manuel de référence de l'API UpdateAccessKey in AWS SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UpdateAccountPasswordPolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UpdateAccountPasswordPolicy`.

CLI

AWS CLI

Pour définir ou modifier la politique de mot de passe du compte actuel

La `update-account-password-policy` commande suivante définit la politique de mot de passe pour exiger une longueur minimale de huit caractères et un ou plusieurs chiffres dans le mot de passe.

```
aws iam update-account-password-policy \
    --minimum-password-length 8 \
    --require-numbers
```

Cette commande ne produit aucun résultat.

Les modifications apportées à la politique de mot de passe d'un compte affectent tous les nouveaux mots de passe créés pour les utilisateurs IAM du compte. Les modifications apportées à la politique des mots de passe n'affectent pas les mots de passe existants.

Pour plus d'informations, consultez la section Définition d'une politique de mot de passe du compte pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateAccountPasswordPolicyà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple met à jour la politique de mot de passe du compte avec les paramètres spécifiés. Notez que les paramètres qui ne sont pas inclus dans la commande ne sont pas laissés inchangés. Au lieu de cela, ils sont réinitialisés aux valeurs par défaut.

```
Update-IAMAccountPasswordPolicy -AllowUsersToChangePasswords $true -HardExpiry
 $false -MaxPasswordAge 90 -MinimumPasswordLength 8 -PasswordReusePrevention 20
 -RequireLowercaseCharacters $true -RequireNumbers $true -RequireSymbols $true -
RequireUppercaseCharacters $true
```

- Pour plus de détails sur l'API, reportez-vous UpdateAccountPasswordPolicyà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UpdateAssumeRolePolicy`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UpdateAssumeRolePolicy`.

CLI

AWS CLI

Pour mettre à jour la politique de confiance pour un rôle IAM

La `update-assume-role-policy` commande suivante met à jour la politique de confiance pour le rôle nomméTest-Role.

```
aws iam update-assume-role-policy \
    --role-name Test-Role \
    --policy-document file://Test-Role-Trust-Policy.json
```

Cette commande ne produit aucun résultat.

La politique d'approbation est définie sous la forme d'un document JSON dans le fichier Test-Role-Trust-Policy.json. (Le nom et l'extension du fichier n'ont aucune importance.) La politique d'approbation doit spécifier un principal.

Pour mettre à jour la politique d'autorisation pour un rôle, utilisez la `put-role-policy` commande.

Pour plus d'informations, consultez Création de rôles IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir UpdateAssumeRolePolicyla section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple met à jour le rôle IAM nommé **`ClientRole`** avec une nouvelle politique de confiance dont le contenu provient du fichier**`ClientRolePolicy.json`**. Notez que vous devez utiliser le paramètre **`-Raw`** switch pour traiter correctement le contenu du fichier JSON.

```
Update-IAMAssumeRolePolicy -RoleName ClientRole -PolicyDocument (Get-Content -raw
 ClientRolePolicy.json)
```

- Pour plus de détails sur l'API, consultez la section UpdateAssumeRolePolicyRéférence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UpdateGroup`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UpdateGroup`.

CLI

AWS CLI

Pour renommer un groupe IAM

La `update-group` commande suivante change le nom du groupe IAM `Test` en`Test-1`.

```
aws iam update-group \
    --group-name Test \
    --new-group-name Test-1
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Affectation d'un nouveau nom à un groupe IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateGroupà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renomme le groupe **`Testers`** IAM en. **`AppTesters`**

```
Update-IAMGroup -GroupName Testers -NewGroupName AppTesters
```

Exemple 2 : Dans cet exemple, le chemin du groupe **`AppTesters`** IAM est remplacé par. **`/Org1/Org2/`** Cela change l'ARN du groupe en**`arn:aws:iam::123456789012:group/Org1/Org2/AppTesters`**.

```
Update-IAMGroup -GroupName AppTesters -NewPath /Org1/Org2/
```

- Pour plus de détails sur l'API, reportez-vous UpdateGroupà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UpdateLoginProfile`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UpdateLoginProfile`.

CLI

AWS CLI

Pour mettre à jour le mot de passe d'un utilisateur IAM

La `update-login-profile` commande suivante crée un nouveau mot de passe pour l'utilisateur IAM nomméBob.

```
aws iam update-login-profile \
    --user-name Bob \
    --password <password>
```

Cette commande ne produit aucun résultat.

Pour définir une politique de mot de passe pour le compte, utilisez la `update-account-password-policy` commande. Si le nouveau mot de passe enfreint la politique de mot de passe du compte, la commande renvoie une `PasswordPolicyViolation` erreur.

Si la politique de mot de passe du compte le permet, les utilisateurs IAM peuvent modifier leurs propres mots de passe à l'aide de la `change-password` commande.

Conservez le mot de passe dans un endroit sûr. Si le mot de passe est perdu, il ne peut pas être récupéré et vous devez en créer un nouveau à l'aide de la `create-login-profile` commande.

Pour plus d'informations, consultez la section Gestion des mots de passe pour les utilisateurs IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section UpdateLoginProfil dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple définit un nouveau mot de passe temporaire pour l'utilisateur **Bob** IAM et demande à ce dernier de le modifier lors de sa prochaine connexion.

```
Update-IAMLoginProfile -UserName Bob -Password "P@ssw0rd1234" -
PasswordResetRequired $true
```

- Pour plus de détails sur l'API, consultez la section UpdateLoginProfil dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UpdateOpenIdConnectProviderThumbprint`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UpdateOpenIdConnectProviderThumbprint`.

CLI

AWS CLI

Pour remplacer la liste existante des empreintes numériques des certificats de serveur par une nouvelle liste

Cet exemple met à jour la liste des empreintes numériques des certificats pour le fournisseur OIDC dont l'ARN doit `arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com` utiliser une nouvelle empreinte numérique.

```
aws iam update-open-id-connect-provider-thumbprint \
    --open-id-connect-provider-arn arn:aws:iam::123456789012:oidc-provider/
example.oidcprovider.com \
    --thumbprint-list 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Création de fournisseurs d'identité OpenID Connect (OIDC) dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateOpenIdConnectProviderThumbprintà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple met à jour la liste des empreintes numériques des certificats pour le fournisseur OIDC dont l'ARN doit **`arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com`** utiliser une nouvelle empreinte numérique. Le fournisseur OIDC partage la nouvelle valeur lorsque le certificat associé au fournisseur change.

```
Update-IAMOpenIDConnectProviderThumbprint -OpenIDConnectProviderArn
 arn:aws:iam::123456789012:oidc-provider/example.oidcprovider.com -ThumbprintList
 7359755EXAMPLEabc3060bce3EXAMPLEec4542a3
```

- Pour plus de détails sur l'API, reportez-vous UpdateOpenIdConnectProviderThumbprintà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UpdateRole`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UpdateRole`.

CLI

AWS CLI

Pour modifier la description ou la durée de session d'un rôle IAM

La `update-role` commande suivante remplace la description du rôle `production-role` IAM `Main production role` et définit la durée maximale de session à 12 heures.

```
aws iam update-role \
```

```
    --role-name production-role \
    --description 'Main production role' \
    --max-session-duration 43200
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez Modification d'un rôle dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateRoleà la section Référence des AWS CLI commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple met à jour la description du rôle et la durée maximale de session (en secondes) pour laquelle la session d'un rôle peut être demandée.

```
Update-IAMRole -RoleName MyRoleName -Description "My testing role" -
MaxSessionDuration 43200
```

- Pour plus de détails sur l'API, reportez-vous UpdateRoleà la section Référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UpdateRoleDescription`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UpdateRoleDescription`.

CLI

AWS CLI

Pour modifier la description d'un rôle IAM

La `update-role` commande suivante remplace la description du rôle `production-role` IAM par. `Main production role`

```
aws iam update-role-description \
    --role-name production-role \
    --description 'Main production role'
```

Sortie :

```
{
    "Role": {
        "Path": "/",
        "RoleName": "production-role",
        "RoleId": "AROA1234567890EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/production-role",
        "CreateDate": "2017-12-06T17:16:37+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole",
                    "Condition": {}
                }
            ]
        },
        "Description": "Main production role"
    }
}
```

Pour plus d'informations, consultez Modification d'un rôle dans le AWS Guide de l'utilisateur IAM.

- Pour plus de détails sur l'API, reportez-vous à UpdateRolela section Description dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple met à jour la description d'un rôle IAM dans votre compte.

```
Update-IAMRoleDescription -RoleName MyRoleName -Description "My testing role"
```

- Pour plus de détails sur l'API, consultez la section UpdateRoleDescription dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UpdateSamlProvider`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UpdateSamlProvider`.

CLI

AWS CLI

Pour mettre à jour le document de métadonnées d'un fournisseur SAML existant

Cet exemple met à jour le fournisseur SAML dans IAM dont l'ARN est `arn:aws:iam::123456789012:saml-provider/SAMLADFS` associé à un nouveau document de métadonnées SAML issu du fichier. `SAMLMetaData.xml`

```
aws iam update-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFS
```

Sortie :

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/SAMLADFS"
}
```

Pour plus d'informations, consultez Création de fournisseurs d'identité SAML IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir UpdateSamlProvider dans AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple met à jour le fournisseur SAML dans IAM dont l'ARN est **`arn:aws:iam::123456789012:saml-provider/SAMLADFS`** associé à un nouveau document de métadonnées SAML issu du fichier. **`SAMLMetaData.xml`** Notez que vous devez utiliser le paramètre **`-Raw`** switch pour traiter correctement le contenu du fichier JSON.

```
Update-IAMSAMLProvider -SAMLProviderArn arn:aws:iam::123456789012:saml-provider/
SAMLADFS -SAMLMetadataDocument (Get-Content -Raw SAMLMetaData.xml)
```

- Pour plus de détails sur l'API, consultez la section UpdateSamlFournisseur dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UpdateServerCertificate`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UpdateServerCertificate`.

C++

Kit de développement logiciel (SDK) for C++

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::updateServerCertificate(const Aws::String
 &currentCertificateName,
                                           const Aws::String &newCertificateName,
                                           const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
```

```
    Aws::IAM::Model::UpdateServerCertificateRequest request;
    request.SetServerCertificateName(currentCertificateName);
    request.SetNewServerCertificateName(newCertificateName);

    auto outcome = iam.UpdateServerCertificate(request);
    bool result = true;
    if (outcome.IsSuccess()) {
        std::cout << "Server certificate " << currentCertificateName
                  << " successfully renamed as " << newCertificateName
                  << std::endl;
    }
    else {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error changing name of server certificate " <<
                      currentCertificateName << " to " << newCertificateName <<
 ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << currentCertificateName
                      << "' not found." << std::endl;
        }
    }

    return result;
}
```

- Pour plus de détails sur l'API, consultez la section UpdateServerCertificat dans la référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour modifier le chemin ou le nom d'un certificat de serveur dans votre AWS compte

La commande `update-server-certificate` suivante fait passer le nom du certificat de `myServerCertificate` à `myUpdatedServerCertificate`. Il modifie également le chemin pour `/cloudfront/` qu'il soit accessible par le CloudFront service Amazon. Cette

commande ne produit aucun résultat. Vous pouvez afficher les résultats de la mise à jour en exécutant la commande `list-server-certificates`.

```
aws-iam update-server-certificate \
    --server-certificate-name myServerCertificate \
    --new-server-certificate-name myUpdatedServerCertificate \
    --new-path /cloudfront/
```

Cette commande ne produit aucun résultat.

Pour de plus amples informations, veuillez consulter Gestion des certificats de serveur dans IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez la section UpdateServerCertificat dans AWS CLI la référence des commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Mettez à jour un certificat de serveur.

```
import { UpdateServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentName
 * @param {string} newName
 */
export const updateServerCertificate = (currentName, newName) => {
  const command = new UpdateServerCertificateCommand({
    ServerCertificateName: currentName,
    NewServerCertificateName: newName,
  });
```

```
  return client.send(command);
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, consultez la section UpdateServerCertificat dans la référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  ServerCertificateName: "CERTIFICATE_NAME",
  NewServerCertificateName: "NEW_CERTIFICATE_NAME",
};

iam.updateServerCertificate(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.

- Pour plus de détails sur l'API, consultez la section UpdateServerCertificat dans la référence des AWS SDK for JavaScript API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renomme le certificat nommé **MyServerCertificate** en**MyRenamedServerCertificate**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -
NewServerCertificateName MyRenamedServerCertificate
```

Exemple 2 : Cet exemple déplace le certificat nommé **MyServerCertificate** vers le chemin /Org1/Org2/. Cela change l'ARN de la ressource en**arn:aws:iam::123456789012:server-certificate/Org1/Org2/MyServerCertificate**.

```
Update-IAMServerCertificate -ServerCertificateName MyServerCertificate -NewPath /
Org1/Org2/
```

- Pour plus de détails sur l'API, consultez la section UpdateServerCertificat dans la référence des AWS Tools for PowerShell applets de commande.

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Répertoriez, mettez à jour et supprimez les certificats de serveur.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
```

```
    @logger = logger
    @logger.progname = "ServerCertificateManager"
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key,
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info("No server certificates found.")
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
#{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
```

```
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Pour plus de détails sur l'API, consultez la section UpdateServerCertificat dans la référence des AWS SDK for Ruby API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UpdateSigningCertificate`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UpdateSigningCertificate`.

CLI

AWS CLI

Pour activer ou désactiver un certificat de signature pour un utilisateur IAM

La `update-signing-certificate` commande suivante désactive le certificat de signature spécifié pour l'utilisateur IAM nommé. Bob

```
aws iam update-signing-certificate \
    --certificate-id TA7SMP42TDN5Z26OBPJE7EXAMPLE \
```

```
    --status Inactive \
    --user-name Bob
```

Pour obtenir l'ID d'un certificat de signature, utilisez la `list-signing-certificates` commande.

Pour plus d'informations, consultez la section Gérer les certificats de signature dans le guide de l'utilisateur Amazon EC2.

- Pour plus de détails sur l'API, consultez la section UpdateSigningCertificat dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple met à jour le certificat associé à l'utilisateur IAM nommé **Bob** et dont l'ID de certificat est destiné **Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU** à le marquer comme inactif.

```
Update-IAMSigningCertificate -CertificateId Y3EK7RMEXAMPLESV33FCREXAMPLEMJLU -
UserName Bob -Status Inactive
```

- Pour plus de détails sur l'API, consultez la section UpdateSigningCertificat dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UpdateUser`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UpdateUser`.

Les exemples d’actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l’exemple de code suivant :

- Créer des utilisateurs en lecture seule et en lecture-écriture

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::IAM::updateUser(const Aws::String &currentUserName,
                             const Aws::String &newUserName,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::UpdateUserRequest request;
    request.SetUserName(currentUserName);
    request.SetNewUserName(newUserName);

    auto outcome = iam.UpdateUser(request);
    if (outcome.IsSuccess()) {
        std::cout << "IAM user " << currentUserName <<
                  " successfully updated with new user name " << newUserName <<
                  std::endl;
    }
    else {
        std::cerr << "Error updating user name for IAM user " << currentUserName
 <<
                  ":" << outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous UpdateUserà la section Référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour modifier le nom d'un utilisateur IAM

La commande `update-user` suivante fait passer le nom de l'utilisateur IAM de `Bob` à `Robert`.

```
aws iam update-user \
    --user-name Bob \
    --new-user-name Robert
```

Cette commande ne produit aucun résultat.

Pour plus d'informations, consultez la section Affectation d'un nouveau nom à un groupe IAM dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, reportez-vous UpdateUserà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.UpdateUserRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
```

```
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <curName> <newName>\s

                Where:
                    curName - The current user name.\s
                    newName - An updated user name.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String curName = args[0];
        String newName = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateIAMUser(iam, curName, newName);
        System.out.println("Done");
        iam.close();
    }

    public static void updateIAMUser(IamClient iam, String curName, String
 newName) {
        try {
            UpdateUserRequest request = UpdateUserRequest.builder()
                    .userName(curName)
                    .newUserName(newName)
                    .build();

            iam.updateUser(request);
            System.out.printf("Successfully updated user to username %s",
 newName);
```

```
        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous UpdateUserà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Mettez à jour l'utilisateur.

```
import { UpdateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentUserName
 * @param {string} newUserName
 */
export const updateUser = (currentUserName, newUserName) => {
  const command = new UpdateUserCommand({
    UserName: currentUserName,
    NewUserName: newUserName,
  });

  return client.send(command);
};
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, reportez-vous UpdateUserà la section Référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
  NewUserName: process.argv[3],
};

iam.updateUser(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for JavaScript.
- Pour plus de détails sur l'API, reportez-vous UpdateUserà la section Référence des AWS SDK for JavaScript API.

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
suspend fun updateIAMUser(
    curName: String?,
    newName: String?
) {
    val request =
        UpdateUserRequest {
            userName = curName
            newUserName = newName
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.updateUser(request)
        println("Successfully updated user to $newName")
    }
}
```

- Pour plus de détails sur l'API, reportez-vous UpdateUserà la section AWS SDK pour la référence de l'API Kotlin.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple renomme l'utilisateur **Bob** IAM en. **Robert**

```
Update-IAMUser -UserName Bob -NewUserName Robert
```

Exemple 2 : Cet exemple modifie le chemin de l'utilisateur IAM **Bob** en**/Org1/Org2/**, ce qui change effectivement l'ARN de l'utilisateur. **arn:aws:iam::123456789012:user/Org1/Org2/bob**

```
Update-IAMUser -UserName Bob -NewPath /Org1/Org2/
```

- Pour plus de détails sur l'API, reportez-vous UpdateUserà la section Référence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def update_user(user_name, new_user_name):
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user
```

- Pour plus de détails sur l'API, consultez UpdateUserle AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
# Updates an IAM user's name
#
# @param current_name [String] The current name of the user
# @param new_name [String] The new name of the user
def update_user_name(current_name, new_name)
  @iam_client.update_user(user_name: current_name, new_user_name: new_name)
  true
rescue StandardError => e
  @logger.error("Error updating user name from '#{current_name}' to
'#{new_name}': #{e.message}")
  false
end
```

- Pour plus de détails sur l'API, reportez-vous UpdateUserà la section Référence des AWS SDK for Ruby API.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UploadServerCertificate`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UploadServerCertificate`.

CLI

AWS CLI

Pour télécharger un certificat de serveur sur votre AWS compte

La commande upload-server-certificate suivante télécharge un certificat de serveur sur votre compte. AWS Dans cet exemple, le certificat se trouve dans le fichier `public_key_cert_file.pem`, la clé privée associée se trouve dans le fichier `my_private_key.pem` et la chaîne de certificats fournie par l'autorité de certification (CA) se trouve dans le fichier `my_certificate_chain_file.pem`. Lorsque le téléchargement du fichier est terminé, il est disponible sous le nom my ServerCertificate. Les paramètres commençant par `file://` indiquent à la commande de lire le contenu du fichier et de l'utiliser comme valeur de paramètre au lieu du nom du fichier lui-même.

```
aws iam upload-server-certificate \
    --server-certificate-name myServerCertificate \
    --certificate-body file://public_key_cert_file.pem \
    --private-key file://my_private_key.pem \
    --certificate-chain file://my_certificate_chain_file.pem
```

Sortie :

```
{
    "ServerCertificateMetadata": {
        "Path": "/",
        "ServerCertificateName": "myServerCertificate",
        "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
        "Arn": "arn:aws:iam::1234567989012:server-certificate/
myServerCertificate",
        "UploadDate": "2019-04-22T21:13:44+00:00",
        "Expiration": "2019-10-15T22:23:16+00:00"
    }
}
```

Pour de plus amples informations, veuillez consulter Creating, Uploading, and Deleting Server Certificates dans le guide d'utilisation d'IAM.

- Pour plus de détails sur l'API, consultez la section UploadServerCertificat dans AWS CLI la référence des commandes.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import { UploadServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "fs";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";
import * as path from "path";

const client = new IAMClient({});

const certMessage = `Generate a certificate and key with the following command,
 or the equivalent for your system.

openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 -nodes \
-keyout example.key -out example.crt -subj "/CN=example.com" \
-addext "subjectAltName=DNS:example.com,DNS:www.example.net,IP:10.0.0.1"
`;

const getCertAndKey = () => {
  try {
    const cert = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.crt"),
    );
    const key = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.key"),
    );
    return { cert, key };
  } catch (err) {
    if (err.code === "ENOENT") {
      throw new Error(
        `Certificate and/or private key not found. ${certMessage}`,
      );
    }

    throw err;
```

```
  }
};

/**
 *
 * @param {string} certificateName
 */
export const uploadServerCertificate = (certificateName) => {
  const { cert, key } = getCertAndKey();
  const command = new UploadServerCertificateCommand({
    ServerCertificateName: certificateName,
    CertificateBody: cert.toString(),
    PrivateKey: key.toString(),
  });

  return client.send(command);
};
```

- Pour plus de détails sur l'API, consultez la section UploadServerCertificat dans la référence des AWS SDK for JavaScript API.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple télécharge un nouveau certificat de serveur sur le compte IAM. Les fichiers contenant le corps du certificat, la clé privée et (éventuellement) la chaîne de certificats doivent tous être codés au format PEM. Notez que les paramètres nécessitent le contenu réel des fichiers plutôt que les noms de fichiers. Vous devez utiliser le paramètre **`-Raw`** switch pour traiter correctement le contenu du fichier.

```
Publish-IAMServerCertificate -ServerCertificateName MyTestCert -CertificateBody
 (Get-Content -Raw server.crt) -PrivateKey (Get-Content -Raw server.key)
```

Sortie :

```
Arn                  : arn:aws:iam::123456789012:server-certificate/MyTestCert
Expiration           : 1/14/2018 9:52:36 AM
Path                 : /
ServerCertificateId  : ASCAJIEXAMPLE7J7HQZYW
```

```
ServerCertificateName : MyTestCert
UploadDate            : 4/21/2015 11:14:16 AM
```

- Pour plus de détails sur l'API, consultez la section UploadServerCertificat dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`UploadSigningCertificate`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`UploadSigningCertificate`.

CLI

AWS CLI

Pour télécharger un certificat de signature pour un utilisateur IAM

La `upload-signing-certificate` commande suivante télécharge un certificat de signature pour l'utilisateur IAM nommé. Bob

```
aws iam upload-signing-certificate \
    --user-name Bob \
    --certificate-body file://certificate.pem
```

Sortie :

```
{
    "Certificate": {
        "UserName": "Bob",
        "Status": "Active",
        "CertificateBody": "-----BEGIN CERTIFICATE-----<certificate-body>-----END
 CERTIFICATE-----",
        "CertificateId": "TA7SMP42TDN5Z26OBPJE7EXAMPLE",
        "UploadDate": "2013-06-06T21:40:08.121Z"
    }
}
```

Le certificat se trouve dans un fichier nommé certificate.pem au format PEM.

Pour plus d'informations, consultez la section Création et téléchargement d'un certificat de signature utilisateur dans le guide d'utilisation d'IAM.

- Pour plus de détails sur l'API, consultez la section UploadSigningCertificat dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Cet exemple télécharge un nouveau certificat de signature X.509 et l'associe à l'utilisateur IAM nommé. **Bob** Le fichier contenant le corps du certificat est codé PEM. Le **CertificateBody** paramètre nécessite le contenu réel du fichier de certificat plutôt que le nom du fichier. Vous devez utiliser le paramètre **-Raw** switch pour traiter correctement le fichier.

```
Publish-IAMSigningCertificate -UserName Bob -CertificateBody (Get-Content -Raw
 SampleSigningCert.pem)
```

Sortie :

```
CertificateBody : -----BEGIN CERTIFICATE-----

 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC

 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6

 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd

 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN

 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD

 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z

 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
                  YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn
+a4GmWIWJ
                  21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/
f0wYK8m9T
                  rDHudUZg3qX4waLG5M43q7Wgc/
MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
```

```
 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
                  nUhVVxYUntneD9+h8Mg9q6q
+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb

 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
                  NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
                  -----END CERTIFICATE-----
CertificateId   : Y3EK7RMEXAMPLESV33FCEXAMPLEHMJLU
Status          : Active
UploadDate      : 4/20/2015 1:26:01 PM
UserName        : Bob
```

- Pour plus de détails sur l'API, consultez la section UploadSigningCertificat dans la référence des AWS Tools for PowerShell applets de commande.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Scénarios pour IAM utilisant des SDK AWS

Les exemples de code suivants vous montrent comment implémenter des scénarios courants dans IAM avec des AWS SDK. Ces scénarios vous montrent comment accomplir des tâches spécifiques en appelant plusieurs fonctions dans IAM. Chaque scénario inclut un lien vers GitHub, où vous pouvez trouver des instructions sur la façon de configurer et d'exécuter le code.

Exemples

- Créez et gérez un service résilient à l'aide d'un AWS SDK
- Création d'un groupe IAM et ajout d'un utilisateur au groupe à l'aide d'un SDK AWS
- Créez un utilisateur IAM et jouez un rôle dans AWS STS l'utilisation d'un SDK AWS
- Créez des utilisateurs IAM en lecture seule et en lecture-écriture à l'aide d'un SDK AWS
- Gérer les clés d'accès IAM à l'aide d'un SDK AWS
- Gérer les politiques IAM à l'aide d'un SDK AWS
- Gérer les rôles IAM à l'aide d'un SDK AWS
- Gérez votre compte IAM à l'aide d'un SDK AWS
- Restaurer une version d'une politique IAM à l'aide d'un SDK AWS

- Utiliser l'API IAM Policy Builder à l'aide d'un SDK AWS

Créez et gérez un service résilient à l'aide d'un AWS SDK

Les exemples de code suivant montrent comment créer un service web à charge équilibrée qui renvoie des recommandations de livres, de films et de chansons. L'exemple montre comment le service répond aux défaillances et comment le restructurer pour accroître la résilience en cas de défaillance.

- Utilisez un groupe Amazon EC2 Auto Scaling pour créer des instances Amazon Elastic Compute Cloud (Amazon EC2) sur la base d'un modèle de lancement et pour maintenir le nombre d'instances dans une plage spécifiée.
- Gérez et distribuez les requêtes HTTP avec Elastic Load Balancing.
- Surveillez l'état des instances d'un groupe Auto Scaling et transférez les demandes uniquement aux instances saines.
- Exécutez un serveur Web Python sur chaque instance EC2 pour gérer les requêtes HTTP. Le serveur Web répond par des recommandations et des surveillances de l'état.
- Simulez un service de recommandation avec une table Amazon DynamoDB.
- Contrôlez la réponse du serveur Web aux demandes et aux contrôles de santé en mettant à jour AWS Systems Manager les paramètres.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Exécutez un scénario interactif à une invite de commande.

```
static async Task Main(string[] args)
{
    _configuration = new ConfigurationBuilder()
        .SetBasePath(Directory.GetCurrentDirectory())
```

```
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally, load local settings.
            .Build();


        // Set up dependency injection for the AWS services.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonIdentityManagementService>()
                    .AddAWSService<IAmazonDynamoDB>()
                    .AddAWSService<IAmazonElasticLoadBalancingV2>()
                    .AddAWSService<IAmazonSimpleSystemsManagement>()
                    .AddAWSService<IAmazonAutoScaling>()
                    .AddAWSService<IAmazonEC2>()
                    .AddTransient<AutoScalerWrapper>()
                    .AddTransient<ElasticLoadBalancerWrapper>()
                    .AddTransient<SmParameterWrapper>()
                    .AddTransient<Recommendations>()
                    .AddSingleton<IConfiguration>(_configuration)
            )
            .Build();

        ServicesSetup(host);
        ResourcesSetup();

        try
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Welcome to the Resilient Architecture Example
Scenario.");
            Console.WriteLine(new string('-', 80));
            await Deploy(true);

            Console.WriteLine("Now let's begin the scenario.");
            Console.WriteLine(new string('-', 80));
            await Demo(true);
```

```
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Finally, let's clean up our resources.");
            Console.WriteLine(new string('-', 80));

            await DestroyResources(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Resilient Architecture Example Scenario is
complete.");
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine($"There was a problem running the scenario:
{ex.Message}");
            await DestroyResources(true);
            Console.WriteLine(new string('-', 80));
        }
    }

    /// <summary>
    /// Setup any common resources, also used for integration testing.
    /// </summary>
    public static void ResourcesSetup()
    {
        _httpClient = new HttpClient();
    }

    /// <summary>
    /// Populate the services for use within the console application.
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _elasticLoadBalancerWrapper =
host.Services.GetRequiredService<ElasticLoadBalancerWrapper>();
        _iamClient =
host.Services.GetRequiredService<IAmazonIdentityManagementService>();
        _recommendations = host.Services.GetRequiredService<Recommendations>();
        _autoScalerWrapper =
host.Services.GetRequiredService<AutoScalerWrapper>();
        _smParameterWrapper =
host.Services.GetRequiredService<SmParameterWrapper>();
```

```
    }

    /// <summary>
    /// Deploy necessary resources for the scenario.
    /// </summary>
    /// <param name="interactive">True to run as interactive.</param>
    /// <returns>True if successful.</returns>
    public static async Task<bool> Deploy(bool interactive)
    {
        var protocol = "HTTP";
        var port = 80;
        var sshPort = 22;

        Console.WriteLine(
            "\nFor this demo, we'll use the AWS SDK for .NET to create several
AWS resources\n" +
            "to set up a load-balanced web service endpoint and explore some ways
to make it resilient\n" +
            "against various kinds of failures.\n\n" +
            "Some of the resources create by this demo are:\n");

        Console.WriteLine(
            "\t* A DynamoDB table that the web service depends on to provide
book, movie, and song recommendations.");
        Console.WriteLine(
            "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server.");
        Console.WriteLine(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones.");
        Console.WriteLine(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests.");
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to start deploying
resources.");
        if (interactive)
            Console.ReadLine();

        // Create and populate the DynamoDB table.
        var databaseTableName = _configuration["databaseName"];
        var recommendationsPath = Path.Join(_configuration["resourcePath"],
            "recommendations_objects.json");
```

```
        Console.WriteLine($"Creating and populating a DynamoDB table named
{databaseTableName}.");
        await _recommendations.CreateDatabaseWithName(databaseTableName);
        await _recommendations.PopulateDatabase(databaseTableName,
recommendationsPath);
        Console.WriteLine(new string('-', 80));

        // Create the EC2 Launch Template.

        Console.WriteLine(
            $"Creating an EC2 launch template that runs
'server_startup_script.sh' when an instance starts.\n"
            + "\nThis script starts a Python web server defined in the
`server.py` script. The web server\n"
            + "listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.\n"
            + "For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
            + "run a web server, such as Apache, with least-privileged
credentials.");
        Console.WriteLine(
            "\nThe template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            + "permissions to access the DynamoDB recommendation table and
Systems Manager parameters\n"
            + "that control the flow of the demo.");

        var startupScriptPath = Path.Join(_configuration["resourcePath"],
            "server_startup_script.sh");
        var instancePolicyPath = Path.Join(_configuration["resourcePath"],
            "instance_policy.json");
        await _autoScalerWrapper.CreateTemplate(startupScriptPath,
instancePolicyPath);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            + "Availability Zone.\n");
        var zones = await _autoScalerWrapper.DescribeAvailabilityZones();
        await _autoScalerWrapper.CreateGroupOfSize(3,
_autoScalerWrapper.GroupName, zones);
        Console.WriteLine(new string('-', 80));
```

```
        Console.WriteLine(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
            + "HTTP requests. You can see these instances in the console or
continue with the demo.\n");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("Creating variables that control the flow of the
demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine(
            "\nCreating an Elastic Load Balancing target group and load balancer.
The target group\n"
            + "defines how the load balancer connects to instances. The load
balancer provides a\n"
            + "single endpoint where clients connect and dispatches requests to
instances in the group.");

        var defaultVpc = await _autoScalerWrapper.GetDefaultVpc();
        var subnets = await
_autoScalerWrapper.GetAllVpcSubnetsForZones(defaultVpc.VpcId, zones);
        var subnetIds = subnets.Select(s => s.SubnetId).ToList();
        var targetGroup = await
_elasticLoadBalancerWrapper.CreateTargetGroupOnVpc(_elasticLoadBalancerWrapper.TargetGro
protocol, port, defaultVpc.VpcId);

        await
_elasticLoadBalancerWrapper.CreateLoadBalancerAndListener(_elasticLoadBalancerWrapper.Lo
subnetIds, targetGroup);
        await
_autoScalerWrapper.AttachLoadBalancerToGroup(_autoScalerWrapper.GroupName,
targetGroup.TargetGroupArn);
        Console.WriteLine("\nVerifying access to the load balancer endpoint...");
        var endPoint = await
_elasticLoadBalancerWrapper.GetEndpointForLoadBalancerByName(_elasticLoadBalancerWrapper
        var loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);

        if (!loadBalancerAccess)
```

```
        {
            Console.WriteLine("\nCouldn't connect to the load balancer, verifying
 that the port is open...");

            var ipString = await _httpClient.GetStringAsync("https://
checkip.amazonaws.com");
            ipString = ipString.Trim();

            var defaultSecurityGroup = await
 _autoScalerWrapper.GetDefaultSecurityGroupForVpc(defaultVpc);
            var portIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, port,
 ipString);
            var sshPortIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, sshPort,
 ipString);

            if (!portIsOpen)
            {
                Console.WriteLine(
                    "\nFor this example to work, the default security group for
 your default VPC must\n"
                    + "allows access from this computer. You can either add it
 automatically from this\n"
                    + "example or add it yourself using the AWS Management
 Console.\n");

                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
 inbound traffic from your computer's IP address?"))
                {
                    await
 _autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, port,
 ipString);
                }
            }

            if (!sshPortIsOpen)
            {
                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
 inbound SSH traffic for debugging from your computer's IP address?"))
                {
```

```
                    await
_autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, sshPort,
ipString);
                }
            }
            loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);
        }

        if (loadBalancerAccess)
        {
            Console.WriteLine("Your load balancer is ready. You can access it by
browsing to:");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        else
        {
            Console.WriteLine(
                "\nCouldn't get a successful response from the load balancer
endpoint. Troubleshoot by\n"
                + "manually verifying that your VPC and security group are
configured correctly and that\n"
                + "you can successfully make a GET request to the load balancer
endpoint:\n");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue with the
demo.");
        if (interactive)
            Console.ReadLine();
        return true;
    }

    /// <summary>
    /// Demonstrate the steps of the scenario.
    /// </summary>
    /// <param name="interactive">True to run as an interactive scenario.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> Demo(bool interactive)
    {
        var ssmOnlyPolicy = Path.Join(_configuration["resourcePath"],
            "ssm_only_policy.json");
```

```
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Resetting parameters to starting values for demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine("\nThis part of the demonstration shows how to toggle
different parts of the system\n" +
                          "to create situations where the web service fails, and
shows how using a resilient\n" +
                          "architecture can keep the web service running in spite
of these failures.");
        Console.WriteLine(new string('-', 88));
        Console.WriteLine("At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine($"The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.\n" +
                          $"The table name is contained in a Systems Manager
parameter named '{_smParameterWrapper.TableParameter}'.\n" +
                          $"To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");
        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as\n" +
                          "healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Instead of failing when the recommendation service
fails, the web service can return a static response.");
        Console.WriteLine("While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.FailureResponseParameter,
"static");

        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a static response.");
```

```
        Console.WriteLine("The service still reports as healthy because health
checks are still shallow.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Let's reinstate the recommendation service.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
_smParameterWrapper.TableName);
        Console.WriteLine(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n" +
            "access the DynamoDB recommendation table.\n"
        );
        await _autoScalerWrapper.CreateInstanceProfileWithName(
            _autoScalerWrapper.BadCredsPolicyName,
            _autoScalerWrapper.BadCredsRoleName,
            _autoScalerWrapper.BadCredsProfileName,
            ssmOnlyPolicy,
            new List<string> { "AmazonSSMManagedInstanceCore" }
        );
        var instances = await
_autoScalerWrapper.GetInstancesByGroupName(_autoScalerWrapper.GroupName);
        var badInstanceId = instances.First();
        var instanceProfile = await
_autoScalerWrapper.GetInstanceProfile(badInstanceId);
        Console.WriteLine(
            $"Replacing the profile for instance {badInstanceId} with a profile
that contains\n" +
            "bad credentials...\n"
        );
        await _autoScalerWrapper.ReplaceInstanceProfile(
            badInstanceId,
            _autoScalerWrapper.BadCredsProfileName,
            instanceProfile.AssociationId
        );
        Console.WriteLine(
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n" +
            "depending on which instance is selected by the load balancer.\n"
        );
        if (interactive)
            await DemoActionChoices();
```

```
        Console.WriteLine("\nLet's implement a deep health check. For this demo,
a deep health check tests whether");
        Console.WriteLine("the web service can access the DynamoDB table that it
depends on for recommendations. Note that");
        Console.WriteLine("the deep health check is only for ELB routing and not
for Auto Scaling instance health.");
        Console.WriteLine("This kind of deep health check is not recommended for
Auto Scaling instance health, because it");
        Console.WriteLine("risks accidental termination of all instances in the
Auto Scaling group when a dependent service fails.");

        Console.WriteLine("\nBy implementing deep health checks, the load
balancer can detect when one of the instances is failing");
        Console.WriteLine("and take that instance out of rotation.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.HealthCheckParameter,
"deep");

        Console.WriteLine($"\nNow, checking target health indicates that the
instance with bad credentials ({badInstanceId})");
        Console.WriteLine("is unhealthy. Note that it might take a minute or two
for the load balancer to detect the unhealthy");
        Console.WriteLine("instance. Sending a GET request to the load balancer
endpoint always returns a recommendation, because");
        Console.WriteLine("the load balancer takes unhealthy instances out of its
rotation.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nBecause the instances in this demo are controlled by
an auto scaler, the simplest way to fix an unhealthy");
        Console.WriteLine("instance is to terminate it and let the auto scaler
start a new instance to replace it.");

        await _autoScalerWrapper.TryTerminateInstanceById(badInstanceId);

        Console.WriteLine($"\nEven while the instance is terminating and the new
instance is starting, sending a GET");
        Console.WriteLine("request to the web service continues to get a
successful recommendation response because");
        Console.WriteLine("starts and reports as healthy, it is included in the
load balancing rotation.");
```

```
        Console.WriteLine("Note that terminating and replacing an instance
typically takes several minutes, during which time you");
        Console.WriteLine("can see the changing health check status until the new
instance is running and healthy.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nIf the recommendation service fails now, deep health
checks mean all instances report as unhealthy.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");

        Console.WriteLine($"\nWhen all instances are unhealthy, the load balancer
continues to route requests even to");
        Console.WriteLine("unhealthy instances, allowing them to fail open and
return a static response rather than fail");
        Console.WriteLine("closed and report failure to the customer.");

        if (interactive)
            await DemoActionChoices();
        await _smParameterWrapper.Reset();

        Console.WriteLine(new string('-', 80));
        return true;
    }

    /// <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="interactive">True to ask the user for cleanup.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> DestroyResources(bool interactive)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine(
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\n" +
            "that were created for this demo."
        );
```

```
        if (!interactive || GetYesNoResponse("Do you want to clean up all demo
 resources? (y/n) "))
        {
            await
 _elasticLoadBalancerWrapper.DeleteLoadBalancerByName(_elasticLoadBalancerWrapper.LoadBal
            await
 _elasticLoadBalancerWrapper.DeleteTargetGroupByName(_elasticLoadBalancerWrapper.TargetGr
            await
 _autoScalerWrapper.TerminateAndDeleteAutoScalingGroupWithName(_autoScalerWrapper.GroupNa
            await
 _autoScalerWrapper.DeleteKeyPairByName(_autoScalerWrapper.KeyPairName);
            await
 _autoScalerWrapper.DeleteTemplateByName(_autoScalerWrapper.LaunchTemplateName);
            await _autoScalerWrapper.DeleteInstanceProfile(
                _autoScalerWrapper.BadCredsProfileName,
                _autoScalerWrapper.BadCredsRoleName
            );
            await
 _recommendations.DestroyDatabaseByName(_recommendations.TableName);
        }
        else
        {
            Console.WriteLine(
                "Ok, we'll leave the resources intact.\n" +
                "Don't forget to delete them when you're done with them or you
 might incur unexpected charges."
            );
        }

        Console.WriteLine(new string('-', 80));
        return true;
    }
```

Créez une classe qui englobe les actions Auto Scaling et Amazon EC2.

```
/// <summary>
/// Encapsulates Amazon EC2 Auto Scaling and EC2 management methods.
/// </summary>
public class AutoScalerWrapper
{
    private readonly IAmazonAutoScaling _amazonAutoScaling;
    private readonly IAmazonEC2 _amazonEc2;
```

```
    private readonly IAmazonSimpleSystemsManagement _amazonSsm;
    private readonly IAmazonIdentityManagementService _amazonIam;

    private readonly string _instanceType = "";
    private readonly string _amiParam = "";
    private readonly string _launchTemplateName = "";
    private readonly string _groupName = "";
    private readonly string _instancePolicyName = "";
    private readonly string _instanceRoleName = "";
    private readonly string _instanceProfileName = "";
    private readonly string _badCredsProfileName = "";
    private readonly string _badCredsRoleName = "";
    private readonly string _badCredsPolicyName = "";
    private readonly string _keyPairName = "";

    public string GroupName => _groupName;
    public string KeyPairName => _keyPairName;
    public string LaunchTemplateName => _launchTemplateName;
    public string InstancePolicyName => _instancePolicyName;
    public string BadCredsProfileName => _badCredsProfileName;
    public string BadCredsRoleName => _badCredsRoleName;
    public string BadCredsPolicyName => _badCredsPolicyName;

    /// <summary>
    /// Constructor for the AutoScalerWrapper.
    /// </summary>
    /// <param name="amazonAutoScaling">The injected AutoScaling client.</param>
    /// <param name="amazonEc2">The injected EC2 client.</param>
    /// <param name="amazonIam">The injected IAM client.</param>
    /// <param name="amazonSsm">The injected SSM client.</param>
    public AutoScalerWrapper(
        IAmazonAutoScaling amazonAutoScaling,
        IAmazonEC2 amazonEc2,
        IAmazonSimpleSystemsManagement amazonSsm,
        IAmazonIdentityManagementService amazonIam,
        IConfiguration configuration)
    {
        _amazonAutoScaling = amazonAutoScaling;
        _amazonEc2 = amazonEc2;
        _amazonSsm = amazonSsm;
        _amazonIam = amazonIam;

        var prefix = configuration["resourcePrefix"];
        _instanceType = configuration["instanceType"];
```

```
        _amiParam = configuration["amiParam"];

        _launchTemplateName = prefix + "-template";
        _groupName = prefix + "-group";
        _instancePolicyName = prefix + "-pol";
        _instanceRoleName = prefix + "-role";
        _instanceProfileName = prefix + "-prof";
        _badCredsPolicyName = prefix + "-bc-pol";
        _badCredsRoleName = prefix + "-bc-role";
        _badCredsProfileName = prefix + "-bc-prof";
        _keyPairName = prefix + "-key-pair";
    }

    /// <summary>
    /// Create a policy, role, and profile that is associated with instances with
a specified name.
    /// An instance's associated profile defines a role that is assumed by the
    /// instance.The role has attached policies that specify the AWS permissions
granted to
    /// clients that run on the instance.
    /// </summary>
    /// <param name="policyName">Name to use for the policy.</param>
    /// <param name="roleName">Name to use for the role.</param>
    /// <param name="profileName">Name to use for the profile.</param>
    /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
    /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
    /// <returns>The Arn of the profile.</returns>
    public async Task<string> CreateInstanceProfileWithName(
        string policyName,
        string roleName,
        string profileName,
        string ssmOnlyPolicyFile,
        List<string>? awsManagedPolicies = null)
    {

        var assumeRoleDoc = "{" +
                                    "\"Version\": \"2012-10-17\"," +
                                    "\"Statement\": [{" +
                                         "\"Effect\": \"Allow\"," +
                                         "\"Principal\": {" +
                                         "\"Service\": [" +
                                             "\"ec2.amazonaws.com\"" +
                                         "]" +
```

```
                                    "}," +
                                "\"Action\": \"sts:AssumeRole\"" +
                                "}]" +
                            "}";

        var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

        var policyArn = "";

        try
        {
            var createPolicyResult = await _amazonIam.CreatePolicyAsync(
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
```

```
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
```

```
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }

    /// <summary>
    /// Create a new key pair and save the file.
    /// </summary>
    /// <param name="newKeyPairName">The name of the new key pair.</param>
    /// <returns>Async task.</returns>
    public async Task CreateKeyPair(string newKeyPairName)
    {
        try
        {
            var keyResponse = await _amazonEc2.CreateKeyPairAsync(
                new CreateKeyPairRequest() { KeyName = newKeyPairName });
            await File.WriteAllTextAsync($"{newKeyPairName}.pem",
                keyResponse.KeyPair.KeyMaterial);
            Console.WriteLine($"Created key pair {newKeyPairName}.");
        }
        catch (AlreadyExistsException)
        {
            Console.WriteLine("Key pair already exists.");
        }
    }

    /// <summary>
    /// Delete the key pair and file by name.
    /// </summary>
    /// <param name="deleteKeyPairName">The key pair to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteKeyPairByName(string deleteKeyPairName)
```

```
    {
        try
        {
            await _amazonEc2.DeleteKeyPairAsync(
                new DeleteKeyPairRequest() { KeyName = deleteKeyPairName });
            File.Delete($"{deleteKeyPairName}.pem");
        }
        catch (FileNotFoundException)
        {
            Console.WriteLine($"Key pair {deleteKeyPairName} not found.");
        }
    }

    /// <summary>
    /// Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.
    /// The launch template specifies a Bash script in its user data field that
runs after
    /// the instance is started. This script installs the Python packages and
starts a Python
    /// web server on the instance.
    /// </summary>
    /// <param name="startupScriptPath">The path to a Bash script file that is
run.</param>
    /// <param name="instancePolicyPath">The path to a permissions policy to
create and attach to the profile.</param>
    /// <returns>The template object.</returns>
    public async Task<Amazon.EC2.Model.LaunchTemplate> CreateTemplate(string
startupScriptPath, string instancePolicyPath)
    {
        await CreateKeyPair(_keyPairName);
        await CreateInstanceProfileWithName(_instancePolicyName,
_instanceRoleName, _instanceProfileName, instancePolicyPath);

        var startServerText = await File.ReadAllTextAsync(startupScriptPath);
        var plainTextBytes = System.Text.Encoding.UTF8.GetBytes(startServerText);

        var amiLatest = await _amazonSsm.GetParameterAsync(
            new GetParameterRequest() { Name = _amiParam });
        var amiId = amiLatest.Parameter.Value;
        var launchTemplateResponse = await _amazonEc2.CreateLaunchTemplateAsync(
            new CreateLaunchTemplateRequest()
            {
                LaunchTemplateName = _launchTemplateName,
```

```
                LaunchTemplateData = new RequestLaunchTemplateData()
                {
                    InstanceType = _instanceType,
                    ImageId = amiId,
                    IamInstanceProfile =
                        new

 LaunchTemplateIamInstanceProfileSpecificationRequest()
                        {
                            Name = _instanceProfileName
                        },
                    KeyName = _keyPairName,
                    UserData = System.Convert.ToBase64String(plainTextBytes)
                }
            });
        return launchTemplateResponse.LaunchTemplate;

    }

    /// <summary>
    /// Get a list of Availability Zones in the AWS Region of the Amazon EC2
 Client.
    /// </summary>
    /// <returns>A list of availability zones.</returns>
    public async Task<List<string>> DescribeAvailabilityZones()
    {
        var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
            new DescribeAvailabilityZonesRequest());
        return zoneResponse.AvailabilityZones.Select(z => z.ZoneName).ToList();
    }

    /// <summary>
    /// Create an EC2 Auto Scaling group of a specified size and name.
    /// </summary>
    /// <param name="groupSize">The size for the group.</param>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="availabilityZones">The availability zones for the group.</
param>
    /// <returns>Async task.</returns>
    public async Task CreateGroupOfSize(int groupSize, string groupName,
 List<string> availabilityZones)
    {
        try
        {
```

```
            await _amazonAutoScaling.CreateAutoScalingGroupAsync(
                new CreateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    AvailabilityZones = availabilityZones,
                    LaunchTemplate =
                        new
Amazon.AutoScaling.Model.LaunchTemplateSpecification()
                        {
                            LaunchTemplateName = _launchTemplateName,
                            Version = "$Default"
                        },
                    MaxSize = groupSize,
                    MinSize = groupSize
                });
            Console.WriteLine($"Created EC2 Auto Scaling group {groupName} with
size {groupSize}.");
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine($"EC2 Auto Scaling group {groupName} already
exists.");
        }
    }

    /// <summary>
    /// Get the default VPC for the account.
    /// </summary>
    /// <returns>The default VPC object.</returns>
    public async Task<Vpc> GetDefaultVpc()
    {
        var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
            new DescribeVpcsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("is-default", new List<string>() { "true" })
                }
            });
        return vpcResponse.Vpcs[0];
    }

    /// <summary>
    /// Get all the subnets for a Vpc in a set of availability zones.
```

```
    /// </summary>
    /// <param name="vpcId">The Id of the Vpc.</param>
    /// <param name="availabilityZones">The list of availability zones.</param>
    /// <returns>The collection of subnet objects.</returns>
    public async Task<List<Subnet>> GetAllVpcSubnetsForZones(string vpcId,
List<string> availabilityZones)
    {
        var subnets = new List<Subnet>();
        var subnetPaginator = _amazonEc2.Paginators.DescribeSubnets(
            new DescribeSubnetsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("vpc-id", new List<string>() { vpcId}),
                    new ("availability-zone", availabilityZones),
                    new ("default-for-az", new List<string>() { "true" })
                }
            });

        // Get the entire list using the paginator.
        await foreach (var subnet in subnetPaginator.Subnets)
        {
            subnets.Add(subnet);
        }

        return subnets;
    }

    /// <summary>
    /// Delete a launch template by name.
    /// </summary>
    /// <param name="templateName">The name of the template to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTemplateByName(string templateName)
    {
        try
        {
            await _amazonEc2.DeleteLaunchTemplateAsync(
                new DeleteLaunchTemplateRequest()
                {
                    LaunchTemplateName = templateName
                });
        }
        catch (AmazonClientException)
```

```
        {
            Console.WriteLine($"Unable to delete template {templateName}.");
        }
    }

    /// <summary>
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
    /// <param name="profileName">The name of the profile to delete.</param>
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
    {
        try
        {
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
```

```
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }

    /// <summary>
    /// Gets data about the instances in an EC2 Auto Scaling group by its group
name.
    /// </summary>
    /// <param name="group">The name of the auto scaling group.</param>
    /// <returns>A collection of instance Ids.</returns>
    public async Task<IEnumerable<string>> GetInstancesByGroupName(string group)
    {
        var instanceResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { group }
            });
        var instanceIds = instanceResponse.AutoScalingGroups.SelectMany(
            g => g.Instances.Select(i => i.InstanceId));
        return instanceIds;
    }

    /// <summary>
    /// Get the instance profile association data for an instance.
    /// </summary>
    /// <param name="instanceId">The Id of the instance.</param>
    /// <returns>Instance profile associations data.</returns>
    public async Task<IamInstanceProfileAssociation> GetInstanceProfile(string
instanceId)
    {
        var response = await
_amazonEc2.DescribeIamInstanceProfileAssociationsAsync(
            new DescribeIamInstanceProfileAssociationsRequest()
            {
```

```
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("instance-id", new List<string>() { instanceId })
                },
            });
        return response.IamInstanceProfileAssociations[0];
    }

    /// <summary>
    /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
    /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
    /// used to restart the Python web server.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to update.</param>
    /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
    /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
    /// <returns>Async task.</returns>
    public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
    {
        await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
        // Allow time before resetting.
        Thread.Sleep(25000);
        var instanceReady = false;
        var retries = 5;
        while (retries-- > 0 && !instanceReady)
        {
            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(10000);
```

```
            var instancesPaginator =
_amazonSsm.Paginators.DescribeInstanceInformation(
                new DescribeInstanceInformationRequest());
            // Get the entire list using the paginator.
            await foreach (var instance in
instancesPaginator.InstanceInformationList)
            {
                instanceReady = instance.InstanceId == instanceId;
                if (instanceReady)
                {
                    break;
                }
            }
        }
        Console.WriteLine($"Sending restart command to instance {instanceId}");
        await _amazonSsm.SendCommandAsync(
            new SendCommandRequest()
            {
                InstanceIds = new List<string>() { instanceId },
                DocumentName = "AWS-RunShellScript",
                Parameters = new Dictionary<string, List<string>>()
                {
                        {"commands", new List<string>() { "cd / && sudo python3
server.py 80" }}
                }
            });
        Console.WriteLine($"Restarted the web server on instance {instanceId}");
    }

    /// <summary>
    /// Try to terminate an instance by its Id.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task TryTerminateInstanceById(string instanceId)
    {
        var stopping = false;
        Console.WriteLine($"Stopping {instanceId}...");
        while (!stopping)
        {
            try
            {
                await
_amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(
```

```
                    new TerminateInstanceInAutoScalingGroupRequest()
                    {
                        InstanceId = instanceId,
                        ShouldDecrementDesiredCapacity = false
                    });
                stopping = true;
            }
            catch (ScalingActivityInProgressException)
            {
                Console.WriteLine($"Scaling activity in progress for
{instanceId}. Waiting...");
                Thread.Sleep(10000);
            }
        }
    }

    /// <summary>
    /// Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
    /// waits and retries until the group is successfully deleted.
    /// </summary>
    /// <param name="groupName">The name of the group to try to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TryDeleteGroupByName(string groupName)
    {
        var stopped = false;
        while (!stopped)
        {
            try
            {
                await _amazonAutoScaling.DeleteAutoScalingGroupAsync(
                    new DeleteAutoScalingGroupRequest()
                    {
                        AutoScalingGroupName = groupName
                    });
                stopped = true;
            }
            catch (Exception e)
                when ((e is ScalingActivityInProgressException)
                      || (e is Amazon.AutoScaling.Model.ResourceInUseException))
            {
                Console.WriteLine($"Some instances are still running.
Waiting...");
                Thread.Sleep(10000);
```

```
            }
        }
    }

    /// <summary>
    /// Terminate instances and delete the Auto Scaling group by name.
    /// </summary>
    /// <param name="groupName">The name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task TerminateAndDeleteAutoScalingGroupWithName(string
groupName)
    {
        var describeGroupsResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { groupName }
            });
        if (describeGroupsResponse.AutoScalingGroups.Any())
        {
            // Update the size to 0.
            await _amazonAutoScaling.UpdateAutoScalingGroupAsync(
                new UpdateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    MinSize = 0
                });
            var group = describeGroupsResponse.AutoScalingGroups[0];
            foreach (var instance in group.Instances)
            {
                await TryTerminateInstanceById(instance.InstanceId);
            }

            await TryDeleteGroupByName(groupName);
        }
        else
        {
            Console.WriteLine($"No groups found with name {groupName}.");
        }
    }


    /// <summary>
    /// Get the default security group for a specified Vpc.
```

```
    /// </summary>
    /// <param name="vpc">The Vpc to search.</param>
    /// <returns>The default security group.</returns>
    public async Task<SecurityGroup> GetDefaultSecurityGroupForVpc(Vpc vpc)
    {
        var groupResponse = await _amazonEc2.DescribeSecurityGroupsAsync(
            new DescribeSecurityGroupsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("group-name", new List<string>() { "default" }),
                    new ("vpc-id", new List<string>() { vpc.VpcId })
                }
            });
        return groupResponse.SecurityGroups[0];
    }

    /// <summary>
    /// Verify the default security group of a Vpc allows ingress from the
calling computer.
    /// This can be done by allowing ingress from this computer's IP address.
    /// In some situations, such as connecting from a corporate network, you must
instead specify
    /// a prefix list Id. You can also temporarily open the port to any IP
address while running this example.
    /// If you do, be sure to remove public access when you're done.
    /// </summary>
    /// <param name="vpc">The group to check.</param>
    /// <param name="port">The port to verify.</param>
    /// <param name="ipAddress">This computer's IP address.</param>
    /// <returns>True if the ip address is allowed on the group.</returns>
    public bool VerifyInboundPortForGroup(SecurityGroup group, int port, string
ipAddress)
    {
        var portIsOpen = false;
        foreach (var ipPermission in group.IpPermissions)
        {
            if (ipPermission.FromPort == port)
            {
                foreach (var ipRange in ipPermission.Ipv4Ranges)
                {
                    var cidr = ipRange.CidrIp;
                    if (cidr.StartsWith(ipAddress) || cidr == "0.0.0.0/0")
                    {
```

```
                        portIsOpen = true;
                    }
                }

                if (ipPermission.PrefixListIds.Any())
                {
                    portIsOpen = true;
                }

                if (!portIsOpen)
                {
                    Console.WriteLine("The inbound rule does not appear to be
open to either this computer's IP\n" +
                                      "address, to all IP addresses (0.0.0.0/0),
or to a prefix list ID.");
                }
                else
                {
                    break;
                }
            }
        }

        return portIsOpen;
    }

    /// <summary>
    /// Add an ingress rule to the specified security group that allows access on
the
    /// specified port from the specified IP address.
    /// </summary>
    /// <param name="groupId">The Id of the security group to modify.</param>
    /// <param name="port">The port to open.</param>
    /// <param name="ipAddress">The IP address to allow access.</param>
    /// <returns>Async task.</returns>
    public async Task OpenInboundPort(string groupId, int port, string ipAddress)
    {
        await _amazonEc2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest()
            {
                GroupId = groupId,
                IpPermissions = new List<IpPermission>()
                {
                    new IpPermission()
```

```
                    {
                        FromPort = port,
                        ToPort = port,
                        IpProtocol = "tcp",
                        Ipv4Ranges = new List<IpRange>()
                        {
                            new IpRange() { CidrIp = $"{ipAddress}/32" }
                        }
                    }
                }
            });
    }

    /// <summary>
    /// Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
 Scaling group.
    /// The
    /// </summary>
    /// <param name="autoScalingGroupName">The name of the Auto Scaling group.</
param>
    /// <param name="targetGroupArn">The Arn for the target group.</param>
    /// <returns>Async task.</returns>
    public async Task AttachLoadBalancerToGroup(string autoScalingGroupName,
 string targetGroupArn)
    {
        await _amazonAutoScaling.AttachLoadBalancerTargetGroupsAsync(
            new AttachLoadBalancerTargetGroupsRequest()
            {
                AutoScalingGroupName = autoScalingGroupName,
                TargetGroupARNs = new List<string>() { targetGroupArn }
            });
    }
}
```

Créez une classe qui englobe les actions Elastic Load Balancing.

```
/// <summary>
/// Encapsulates Elastic Load Balancer actions.
/// </summary>
public class ElasticLoadBalancerWrapper
{
```

```
    private readonly IAmazonElasticLoadBalancingV2 _amazonElasticLoadBalancingV2;
    private string? _endpoint = null;
    private readonly string _targetGroupName = "";
    private readonly string _loadBalancerName = "";
    HttpClient _httpClient = new();

    public string TargetGroupName => _targetGroupName;
    public string LoadBalancerName => _loadBalancerName;

    /// <summary>
    /// Constructor for the Elastic Load Balancer wrapper.
    /// </summary>
    /// <param name="amazonElasticLoadBalancingV2">The injected load balancing v2
client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public ElasticLoadBalancerWrapper(
        IAmazonElasticLoadBalancingV2 amazonElasticLoadBalancingV2,
        IConfiguration configuration)
    {
        _amazonElasticLoadBalancingV2 = amazonElasticLoadBalancingV2;
        var prefix = configuration["resourcePrefix"];
        _targetGroupName = prefix + "-tg";
        _loadBalancerName = prefix + "-lb";
    }

    /// <summary>
    /// Get the HTTP Endpoint of a load balancer by its name.
    /// </summary>
    /// <param name="loadBalancerName">The name of the load balancer.</param>
    /// <returns>The HTTP endpoint.</returns>
    public async Task<string> GetEndpointForLoadBalancerByName(string
loadBalancerName)
    {
        if (_endpoint == null)
        {
            var endpointResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { loadBalancerName }
                    });
            _endpoint = endpointResponse.LoadBalancers[0].DNSName;
        }
```

```
        return _endpoint;
    }

    /// <summary>
    /// Return the GET response for an endpoint as text.
    /// </summary>
    /// <param name="endpoint">The endpoint for the request.</param>
    /// <returns>The request response.</returns>
    public async Task<string> GetEndPointResponse(string endpoint)
    {
        var endpointResponse = await _httpClient.GetAsync($"http://{endpoint}");
        var textResponse = await endpointResponse.Content.ReadAsStringAsync();
        return textResponse!;
    }

    /// <summary>
    /// Get the target health for a group by name.
    /// </summary>
    /// <param name="groupName">The name of the group.</param>
    /// <returns>The collection of health descriptions.</returns>
    public async Task<List<TargetHealthDescription>>
CheckTargetHealthForGroup(string groupName)
    {
        List<TargetHealthDescription> result = null!;
        try
        {
            var groupResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                    new DescribeTargetGroupsRequest()
                    {
                        Names = new List<string>() { groupName }
                    });
            var healthResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetHealthAsync(
                    new DescribeTargetHealthRequest()
                    {
                        TargetGroupArn =
groupResponse.TargetGroups[0].TargetGroupArn
                    });
            ;
            result = healthResponse.TargetHealthDescriptions;
        }
        catch (TargetGroupNotFoundException)
        {
```

```
            Console.WriteLine($"Target group {groupName} not found.");
        }
        return result;
    }

    /// <summary>
    /// Create an Elastic Load Balancing target group. The target group specifies
 how the load balancer forwards
    /// requests to instances in the group and how instance health is checked.
    ///
    /// To speed up this demo, the health check is configured with shortened
 times and lower thresholds. In production,
    /// you might want to decrease the sensitivity of your health checks to avoid
 unwanted failures.
    /// </summary>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="protocol">The protocol, such as HTTP.</param>
    /// <param name="port">The port to use to forward requests, such as 80.</
param>
    /// <param name="vpcId">The Id of the Vpc in which the load balancer
 exists.</param>
    /// <returns>The new TargetGroup object.</returns>
    public async Task<TargetGroup> CreateTargetGroupOnVpc(string groupName,
 ProtocolEnum protocol, int port, string vpcId)
    {
        var createResponse = await
 _amazonElasticLoadBalancingV2.CreateTargetGroupAsync(
            new CreateTargetGroupRequest()
            {
                Name = groupName,
                Protocol = protocol,
                Port = port,
                HealthCheckPath = "/healthcheck",
                HealthCheckIntervalSeconds = 10,
                HealthCheckTimeoutSeconds = 5,
                HealthyThresholdCount = 2,
                UnhealthyThresholdCount = 2,
                VpcId = vpcId
            });
        var targetGroup = createResponse.TargetGroups[0];
        return targetGroup;
    }

    /// <summary>
```

```
   /// Create an Elastic Load Balancing load balancer that uses the specified
subnets
   /// and forwards requests to the specified target group.
   /// </summary>
   /// <param name="name">The name for the new load balancer.</param>
   /// <param name="subnetIds">Subnets for the load balancer.</param>
   /// <param name="targetGroup">Target group for forwarded requests.</param>
   /// <returns>The new LoadBalancer object.</returns>
   public async Task<LoadBalancer> CreateLoadBalancerAndListener(string name,
List<string> subnetIds, TargetGroup targetGroup)
   {
       var createLbResponse = await
_amazonElasticLoadBalancingV2.CreateLoadBalancerAsync(
           new CreateLoadBalancerRequest()
           {
               Name = name,
               Subnets = subnetIds
           });
       var loadBalancerArn = createLbResponse.LoadBalancers[0].LoadBalancerArn;

       // Wait for load balancer to be available.
       var loadBalancerReady = false;
       while (!loadBalancerReady)
       {
           try
           {
               var describeResponse =
                   await
_amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                       new DescribeLoadBalancersRequest()
                       {
                           Names = new List<string>() { name }
                       });

               var loadBalancerState =
describeResponse.LoadBalancers[0].State.Code;

               loadBalancerReady = loadBalancerState ==
LoadBalancerStateEnum.Active;
           }
           catch (LoadBalancerNotFoundException)
           {
               loadBalancerReady = false;
           }
```

```
            Thread.Sleep(10000);
        }
        // Create the listener.
        await _amazonElasticLoadBalancingV2.CreateListenerAsync(
            new CreateListenerRequest()
            {
                LoadBalancerArn = loadBalancerArn,
                Protocol = targetGroup.Protocol,
                Port = targetGroup.Port,
                DefaultActions = new List<Action>()
                {
                    new Action()
                    {
                        Type = ActionTypeEnum.Forward,
                        TargetGroupArn = targetGroup.TargetGroupArn
                    }
                }
            });
        return createLbResponse.LoadBalancers[0];
    }

    /// <summary>
    /// Verify this computer can successfully send a GET request to the
    /// load balancer endpoint.
    /// </summary>
    /// <param name="endpoint">The endpoint to check.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> VerifyLoadBalancerEndpoint(string endpoint)
    {
        var success = false;
        var retries = 3;
        while (!success && retries > 0)
        {
            try
            {
                var endpointResponse = await _httpClient.GetAsync($"http://
{endpoint}");
                Console.WriteLine($"Response: {endpointResponse.StatusCode}.");

                if (endpointResponse.IsSuccessStatusCode)
                {
                    success = true;
                }
                else
```

```
                {
                    retries = 0;
                }
            }
            catch (HttpRequestException)
            {
                Console.WriteLine("Connection error, retrying...");
                retries--;
                Thread.Sleep(10000);
            }
        }

        return success;
    }

    /// <summary>
    /// Delete a load balancer by its specified name.
    /// </summary>
    /// <param name="name">The name of the load balancer to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteLoadBalancerByName(string name)
    {
        try
        {
            var describeLoadBalancerResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { name }
                    });
            var lbArn =
describeLoadBalancerResponse.LoadBalancers[0].LoadBalancerArn;
            await _amazonElasticLoadBalancingV2.DeleteLoadBalancerAsync(
                new DeleteLoadBalancerRequest()
                {
                    LoadBalancerArn = lbArn
                }
            );
        }
        catch (LoadBalancerNotFoundException)
        {
            Console.WriteLine($"Load balancer {name} not found.");
        }
    }
```

```
    /// <summary>
    /// Delete a TargetGroup by its specified name.
    /// </summary>
    /// <param name="groupName">Name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTargetGroupByName(string groupName)
    {
        var done = false;
        while (!done)
        {
            try
            {
                var groupResponse =
                    await
_amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                        new DescribeTargetGroupsRequest()
                        {
                            Names = new List<string>() { groupName }
                        });

                var targetArn = groupResponse.TargetGroups[0].TargetGroupArn;
                await _amazonElasticLoadBalancingV2.DeleteTargetGroupAsync(
                    new DeleteTargetGroupRequest() { TargetGroupArn =
targetArn });
                Console.WriteLine($"Deleted load balancing target group
{groupName}.");
                done = true;
            }
            catch (TargetGroupNotFoundException)
            {
                Console.WriteLine(
                    $"Target group {groupName} not found, could not delete.");
                done = true;
            }
            catch (ResourceInUseException)
            {
                Console.WriteLine("Target group not yet released, waiting...");
                Thread.Sleep(10000);
            }
        }
    }
}
```

Créez une classe qui utilise DynamoDB pour simuler un service de recommandation.

```
/// <summary>
/// Encapsulates a DynamoDB table to use as a service that recommends books,
 movies, and songs.
/// </summary>
public class Recommendations
{
    private readonly IAmazonDynamoDB _amazonDynamoDb;
    private readonly DynamoDBContext _context;
    private readonly string _tableName;

    public string TableName => _tableName;

    /// <summary>
    /// Constructor for the Recommendations service.
    /// </summary>
    /// <param name="amazonDynamoDb">The injected DynamoDb client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public Recommendations(IAmazonDynamoDB amazonDynamoDb, IConfiguration
 configuration)
    {
        _amazonDynamoDb = amazonDynamoDb;
        _context = new DynamoDBContext(_amazonDynamoDb);
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Create the DynamoDb table with a specified name.
    /// </summary>
    /// <param name="tableName">The name for the table.</param>
    /// <returns>True when ready.</returns>
    public async Task<bool> CreateDatabaseWithName(string tableName)
    {
        try
        {
            Console.Write($"Creating table {tableName}...");
            var createRequest = new CreateTableRequest()
            {
                TableName = tableName,
                AttributeDefinitions = new List<AttributeDefinition>()
```

```
                    {
                        new AttributeDefinition()
                        {
                            AttributeName = "MediaType",
                            AttributeType = ScalarAttributeType.S
                        },
                        new AttributeDefinition()
                        {
                            AttributeName = "ItemId",
                            AttributeType = ScalarAttributeType.N
                        }
                    },
                KeySchema = new List<KeySchemaElement>()
                    {
                        new KeySchemaElement()
                        {
                            AttributeName = "MediaType",
                            KeyType = KeyType.HASH
                        },
                        new KeySchemaElement()
                        {
                            AttributeName = "ItemId",
                            KeyType = KeyType.RANGE
                        }
                    },
                ProvisionedThroughput = new ProvisionedThroughput()
                {
                    ReadCapacityUnits = 5,
                    WriteCapacityUnits = 5
                }
            };
            await _amazonDynamoDb.CreateTableAsync(createRequest);

            // Wait until the table is ACTIVE and then report success.
            Console.Write("\nWaiting for table to become active...");

            var request = new DescribeTableRequest
            {
                TableName = tableName
            };

            TableStatus status;
            do
            {
```

```
                Thread.Sleep(2000);

                var describeTableResponse = await
 _amazonDynamoDb.DescribeTableAsync(request);
                status = describeTableResponse.Table.TableStatus;

                Console.Write(".");
            }
            while (status != "ACTIVE");

            return status == TableStatus.ACTIVE;
        }
        catch (ResourceInUseException)
        {
            Console.WriteLine($"Table {tableName} already exists.");
            return false;
        }
    }

    /// <summary>
    /// Populate the database table with data from a specified path.
    /// </summary>
    /// <param name="databaseTableName">The name of the table.</param>
    /// <param name="recommendationsPath">The path of the recommendations data.</
param>
    /// <returns>Async task.</returns>
    public async Task PopulateDatabase(string databaseTableName, string
 recommendationsPath)
    {
        var recommendationsText = await
 File.ReadAllTextAsync(recommendationsPath);
        var records =

 JsonSerializer.Deserialize<RecommendationModel[]>(recommendationsText);
        var batchWrite = _context.CreateBatchWrite<RecommendationModel>();

        foreach (var record in records!)
        {
            batchWrite.AddPutItem(record);
        }

        await batchWrite.ExecuteAsync();
    }
```

```
    /// <summary>
    /// Delete the recommendation table by name.
    /// </summary>
    /// <param name="tableName">The name of the recommendation table.</param>
    /// <returns>Async task.</returns>
    public async Task DestroyDatabaseByName(string tableName)
    {
        try
        {
            await _amazonDynamoDb.DeleteTableAsync(
                new DeleteTableRequest() { TableName = tableName });
            Console.WriteLine($"Table {tableName} was deleted.");
        }
        catch (ResourceNotFoundException)
        {
            Console.WriteLine($"Table {tableName} not found");
        }
    }
}
```

Créez une classe qui englobe les actions de Systems Manager.

```
/// <summary>
/// Encapsulates Systems Manager parameter operations. This example uses these
 parameters
/// to drive the demonstration of resilient architecture, such as failure of a
 dependency or
/// how the service responds to a health check.
/// </summary>
public class SmParameterWrapper
{
    private readonly IAmazonSimpleSystemsManagement
 _amazonSimpleSystemsManagement;

    private readonly string _tableParameter = "doc-example-resilient-
architecture-table";
    private readonly string _failureResponseParameter = "doc-example-resilient-
architecture-failure-response";
    private readonly string _healthCheckParameter = "doc-example-resilient-
architecture-health-check";
    private readonly string _tableName = "";
```

```
    public string TableParameter => _tableParameter;
    public string TableName => _tableName;
    public string HealthCheckParameter => _healthCheckParameter;
    public string FailureResponseParameter => _failureResponseParameter;

    /// <summary>
    /// Constructor for the SmParameterWrapper.
    /// </summary>
    /// <param name="amazonSimpleSystemsManagement">The injected Simple Systems
 Management client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public SmParameterWrapper(IAmazonSimpleSystemsManagement
 amazonSimpleSystemsManagement, IConfiguration configuration)
    {
        _amazonSimpleSystemsManagement = amazonSimpleSystemsManagement;
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Reset the Systems Manager parameters to starting values for the demo.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task Reset()
    {
        await this.PutParameterByName(_tableParameter, _tableName);
        await this.PutParameterByName(_failureResponseParameter, "none");
        await this.PutParameterByName(_healthCheckParameter, "shallow");
    }

    /// <summary>
    /// Set the value of a named Systems Manager parameter.
    /// </summary>
    /// <param name="name">The name of the parameter.</param>
    /// <param name="value">The value to set.</param>
    /// <returns>Async task.</returns>
    public async Task PutParameterByName(string name, string value)
    {
        await _amazonSimpleSystemsManagement.PutParameterAsync(
            new PutParameterRequest() { Name = name, Value = value, Overwrite =
 true });
    }
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK for .NET .
  - AttachLoadBalancerTargetGroupes
  - CreateAutoScalingGroup
  - CreateInstanceProfil
  - CreateLaunchModèle
  - CreateListener
  - CreateLoadÉquilibreur
  - CreateTargetGroupe
  - DeleteAutoScalingGroup
  - DeleteInstanceProfil
  - DeleteLaunchModèle
  - DeleteLoadÉquilibreur
  - DeleteTargetGroupe
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadÉquilibreurs
  - DescribeSubnets
  - DescribeTargetGroupes
  - DescribeTargetHealth
  - DescribeVpcs
  - RebootInstances
  - ReplaceIamInstanceProfileAssociation
  - TerminateInstanceInAutoScalingGroup
  - UpdateAutoScalingGroup

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Exécutez un scénario interactif à une invite de commande.

```
public class Main {

    public static final String fileName = "C:\\AWS\\resworkflow\
\recommendations.json"; // Modify file location.
    public static final String tableName = "doc-example-recommendation-service";
    public static final String startScript = "C:\\AWS\\resworkflow\
\server_startup_script.sh"; // Modify file location.
    public static final String policyFile = "C:\\AWS\\resworkflow\
\instance_policy.json"; // Modify file location.
    public static final String ssmJSON = "C:\\AWS\\resworkflow\
\ssm_only_policy.json"; // Modify file location.
    public static final String failureResponse = "doc-example-resilient-
architecture-failure-response";
    public static final String healthCheck = "doc-example-resilient-architecture-
health-check";
    public static final String templateName = "doc-example-resilience-template";
    public static final String roleName = "doc-example-resilience-role";
    public static final String policyName = "doc-example-resilience-pol";
    public static final String profileName = "doc-example-resilience-prof";

    public static final String badCredsProfileName = "doc-example-resilience-
prof-bc";

    public static final String targetGroupName = "doc-example-resilience-tg";
    public static final String autoScalingGroupName = "doc-example-resilience-
group";
    public static final String lbName = "doc-example-resilience-lb";
    public static final String protocol = "HTTP";
    public static final int port = 80;
```

```
   public static final String DASHES = new String(new char[80]).replace("\0",
"-");

   public static void main(String[] args) throws IOException,
InterruptedException {
       Scanner in = new Scanner(System.in);
       Database database = new Database();
       AutoScaler autoScaler = new AutoScaler();
       LoadBalancer loadBalancer = new LoadBalancer();

       System.out.println(DASHES);
       System.out.println("Welcome to the demonstration of How to Build and
Manage a Resilient Service!");
       System.out.println(DASHES);

       System.out.println(DASHES);
       System.out.println("A - SETUP THE RESOURCES");
       System.out.println("Press Enter when you're ready to start deploying
resources.");
       in.nextLine();
       deploy(loadBalancer);
       System.out.println(DASHES);
       System.out.println(DASHES);
       System.out.println("B - DEMO THE RESILIENCE FUNCTIONALITY");
       System.out.println("Press Enter when you're ready.");
       in.nextLine();
       demo(loadBalancer);
       System.out.println(DASHES);

       System.out.println(DASHES);
       System.out.println("C - DELETE THE RESOURCES");
       System.out.println("""
               This concludes the demo of how to build and manage a resilient
service.
               To keep things tidy and to avoid unwanted charges on your
account, we can clean up all AWS resources
               that were created for this demo.
               """);

       System.out.println("\n Do you want to delete the resources (y/n)? ");
       String userInput = in.nextLine().trim().toLowerCase(); // Capture user
input

       if (userInput.equals("y")) {
```

```
            // Delete resources here
            deleteResources(loadBalancer, autoScaler, database);
            System.out.println("Resources deleted.");
        } else {
            System.out.println("""
                    Okay, we'll leave the resources intact.
                    Don't forget to delete them when you're done with them or you
might incur unexpected charges.
                    """);
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("The example has completed. ");
        System.out.println("\n Thanks for watching!");
        System.out.println(DASHES);
    }

    // Deletes the AWS resources used in this example.
    private static void deleteResources(LoadBalancer loadBalancer, AutoScaler
autoScaler, Database database)
            throws IOException, InterruptedException {
        loadBalancer.deleteLoadBalancer(lbName);
        System.out.println("*** Wait 30 secs for resource to be deleted");
        TimeUnit.SECONDS.sleep(30);
        loadBalancer.deleteTargetGroup(targetGroupName);
        autoScaler.deleteAutoScaleGroup(autoScalingGroupName);
        autoScaler.deleteRolesPolicies(policyName, roleName, profileName);
        autoScaler.deleteTemplate(templateName);
        database.deleteTable(tableName);
    }

    private static void deploy(LoadBalancer loadBalancer) throws
InterruptedException, IOException {
        Scanner in = new Scanner(System.in);
        System.out.println(
                """
                        For this demo, we'll use the AWS SDK for Java (v2) to
create several AWS resources
                        to set up a load-balanced web service endpoint and
explore some ways to make it resilient
                        against various kinds of failures.

                        Some of the resources create by this demo are:
```

```
                        \t* A DynamoDB table that the web service depends on to
provide book, movie, and song recommendations.
                        \t* An EC2 launch template that defines EC2 instances
that each contain a Python web server.
                        \t* An EC2 Auto Scaling group that manages EC2 instances
across several Availability Zones.
                        \t* An Elastic Load Balancing (ELB) load balancer that
targets the Auto Scaling group to distribute requests.
                        """);

        System.out.println("Press Enter when you're ready.");
        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating and populating a DynamoDB table named " +
tableName);
        Database database = new Database();
        database.createTable(tableName, fileName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.
                This script starts a Python web server defined in the `server.py`
script. The web server
                listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.
                For demo purposes, this server is run as the root user. In
production, the best practice is to
                run a web server, such as Apache, with least-privileged
credentials.

                The template also defines an IAM policy that each instance uses
to assume a role that grants
                permissions to access the DynamoDB recommendation table and
Systems Manager parameters
                that control the flow of the demo.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        templateCreator.createTemplate(policyFile, policyName, profileName,
startScript, templateName, roleName);
```

```
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(
                "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different Availability Zone.");
        System.out.println("*** Wait 30 secs for the VPC to be created");
        TimeUnit.SECONDS.sleep(30);
        AutoScaler autoScaler = new AutoScaler();
        String[] zones = autoScaler.createGroup(3, templateName,
autoScalingGroupName);

        System.out.println("""
                At this point, you have EC2 instances created. Once each instance
starts, it listens for
                HTTP requests. You can see these instances in the console or
continue with the demo.
                Press Enter when you're ready to continue.
                """);

        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating variables that control the flow of the
demo.");
        ParameterHelper paramHelper = new ParameterHelper();
        paramHelper.reset();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an Elastic Load Balancing target group and load
balancer. The target group
                defines how the load balancer connects to instances. The load
balancer provides a
                single endpoint where clients connect and dispatches requests to
instances in the group.
                """);

        String vpcId = autoScaler.getDefaultVPC();
        List<Subnet> subnets = autoScaler.getSubnets(vpcId, zones);
        System.out.println("You have retrieved a list with " + subnets.size() + "
subnets");
```

```
        String targetGroupArn = loadBalancer.createTargetGroup(protocol, port,
vpcId, targetGroupName);
        String elbDnsName = loadBalancer.createLoadBalancer(subnets,
targetGroupArn, lbName, port, protocol);
        autoScaler.attachLoadBalancerTargetGroup(autoScalingGroupName,
targetGroupArn);
        System.out.println("Verifying access to the load balancer endpoint...");
        boolean wasSuccessul =
loadBalancer.verifyLoadBalancerEndpoint(elbDnsName);
        if (!wasSuccessul) {
            System.out.println("Couldn't connect to the load balancer, verifying
that the port is open...");
            CloseableHttpClient httpClient = HttpClients.createDefault();

            // Create an HTTP GET request to "http://checkip.amazonaws.com"
            HttpGet httpGet = new HttpGet("http://checkip.amazonaws.com");
            try {
                // Execute the request and get the response
                HttpResponse response = httpClient.execute(httpGet);

                // Read the response content.
                String ipAddress =
IOUtils.toString(response.getEntity().getContent(),
StandardCharsets.UTF_8).trim();

                // Print the public IP address.
                System.out.println("Public IP Address: " + ipAddress);
                GroupInfo groupInfo = autoScaler.verifyInboundPort(vpcId, port,
ipAddress);
                if (!groupInfo.isPortOpen()) {
                    System.out.println("""
                            For this example to work, the default security group
for your default VPC must
                            allow access from this computer. You can either add
it automatically from this
                            example or add it yourself using the AWS Management
Console.
                            """);

                    System.out.println(
                            "Do you want to add a rule to security group " +
groupInfo.getGroupName() + " to allow");
                    System.out.println("inbound traffic on port " + port + " from
your computer's IP address (y/n) ");
```

```
                    String ans = in.nextLine();
                    if ("y".equalsIgnoreCase(ans)) {
                        autoScaler.openInboundPort(groupInfo.getGroupName(),
String.valueOf(port), ipAddress);
                        System.out.println("Security group rule added.");
                    } else {
                        System.out.println("No security group rule added.");
                    }
                }

            } catch (AutoScalingException e) {
                e.printStackTrace();
            }
        } else if (wasSuccessul) {
            System.out.println("Your load balancer is ready. You can access it by
browsing to:");
            System.out.println("\t http://" + elbDnsName);
        } else {
            System.out.println("Couldn't get a successful response from the load
balancer endpoint. Troubleshoot by");
            System.out.println("manually verifying that your VPC and security
group are configured correctly and that");
            System.out.println("you can successfully make a GET request to the
load balancer.");
        }

        System.out.println("Press Enter when you're ready to continue with the
demo.");
        in.nextLine();
    }

    // A method that controls the demo part of the Java program.
    public static void demo(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        ParameterHelper paramHelper = new ParameterHelper();
        System.out.println("Read the ssm_only_policy.json file");
        String ssmOnlyPolicy = readFileAsString(ssmJSON);

        System.out.println("Resetting parameters to starting values for demo.");
        paramHelper.reset();

        System.out.println(
                """
```

```
                        This part of the demonstration shows how to toggle
different parts of the system
                        to create situations where the web service fails, and
shows how using a resilient
                        architecture can keep the web service running in spite
of these failures.

                        At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.
                       """);
       demoChoices(loadBalancer);

       System.out.println(
               """
                        The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.
                        The table name is contained in a Systems Manager
parameter named self.param_helper.table.
                        To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.
                       """);
       paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

       System.out.println(
               """
                        \nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as
                        healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.
                       """);
       demoChoices(loadBalancer);

       System.out.println(
               """
                       Instead of failing when the recommendation service fails,
the web service can return a static response.
                       While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.
                       """);
       paramHelper.put(paramHelper.failureResponse, "static");

       System.out.println("""
               Now, sending a GET request to the load balancer endpoint returns
a static response.
```

```
                The service still reports as healthy because health checks are
still shallow.
                """);
        demoChoices(loadBalancer);

        System.out.println("Let's reinstate the recommendation service.");
        paramHelper.put(paramHelper.tableName, paramHelper.dyntable);

        System.out.println("""
                Let's also substitute bad credentials for one of the instances in
the target group so that it can't
                access the DynamoDB recommendation table. We will get an instance
id value.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        AutoScaler autoScaler = new AutoScaler();

        // Create a new instance profile based on badCredsProfileName.
        templateCreator.createInstanceProfile(policyFile, policyName,
badCredsProfileName, roleName);
        String badInstanceId = autoScaler.getBadInstance(autoScalingGroupName);
        System.out.println("The bad instance id values used for this demo is " +
badInstanceId);

        String profileAssociationId =
autoScaler.getInstanceProfile(badInstanceId);
        System.out.println("The association Id value is " +
profileAssociationId);
        System.out.println("Replacing the profile for instance " + badInstanceId
                + " with a profile that contains bad credentials");
        autoScaler.replaceInstanceProfile(badInstanceId, badCredsProfileName,
profileAssociationId);

        System.out.println(
                """
                        Now, sending a GET request to the load balancer endpoint
returns either a recommendation or a static response,
                        depending on which instance is selected by the load
balancer.
                        """);

        demoChoices(loadBalancer);
```

```
        System.out.println("""
                Let's implement a deep health check. For this demo, a deep health
check tests whether
                the web service can access the DynamoDB table that it depends on
for recommendations. Note that
                the deep health check is only for ELB routing and not for Auto
Scaling instance health.
                This kind of deep health check is not recommended for Auto
Scaling instance health, because it
                risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.
                """);

        System.out.println("""
                By implementing deep health checks, the load balancer can detect
when one of the instances is failing
                and take that instance out of rotation.
                """);

        paramHelper.put(paramHelper.healthCheck, "deep");

        System.out.println("""
                Now, checking target health indicates that the instance with bad
credentials
                is unhealthy. Note that it might take a minute or two for the
load balancer to detect the unhealthy
                instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because
                the load balancer takes unhealthy instances out of its rotation.
                """);

        demoChoices(loadBalancer);

        System.out.println(
                """
                        Because the instances in this demo are controlled by an
auto scaler, the simplest way to fix an unhealthy
                        instance is to terminate it and let the auto scaler start
a new instance to replace it.
                        """);
        autoScaler.terminateInstance(badInstanceId);

        System.out.println("""
```

```
                Even while the instance is terminating and the new instance is
starting, sending a GET
                request to the web service continues to get a successful
recommendation response because
                the load balancer routes requests to the healthy instances. After
the replacement instance
                starts and reports as healthy, it is included in the load
balancing rotation.
                Note that terminating and replacing an instance typically takes
several minutes, during which time you
                can see the changing health check status until the new instance
is running and healthy.
                """);

        demoChoices(loadBalancer);
        System.out.println(
                "If the recommendation service fails now, deep health checks mean
all instances report as unhealthy.");
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        demoChoices(loadBalancer);
        paramHelper.reset();
    }

    public static void demoChoices(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        String[] actions = {
                "Send a GET request to the load balancer endpoint.",
                "Check the health of load balancer targets.",
                "Go to the next part of the demo."
        };
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("-".repeat(88));
            System.out.println("See the current state of the service by selecting
one of the following choices:");
            for (int i = 0; i < actions.length; i++) {
                System.out.println(i + ": " + actions[i]);
            }

            try {
                System.out.print("\nWhich action would you like to take? ");
                int choice = scanner.nextInt();
```

```
                System.out.println("-".repeat(88));

                switch (choice) {
                    case 0 -> {
                        System.out.println("Request:\n");
                        System.out.println("GET http://" +
loadBalancer.getEndpoint(lbName));
                        CloseableHttpClient httpClient =
HttpClients.createDefault();

                        // Create an HTTP GET request to the ELB.
                        HttpGet httpGet = new HttpGet("http://" +
loadBalancer.getEndpoint(lbName));

                        // Execute the request and get the response.
                        HttpResponse response = httpClient.execute(httpGet);
                        int statusCode =
response.getStatusLine().getStatusCode();
                        System.out.println("HTTP Status Code: " + statusCode);

                        // Display the JSON response
                        BufferedReader reader = new BufferedReader(
                                new
InputStreamReader(response.getEntity().getContent()));
                        StringBuilder jsonResponse = new StringBuilder();
                        String line;
                        while ((line = reader.readLine()) != null) {
                            jsonResponse.append(line);
                        }
                        reader.close();

                        // Print the formatted JSON response.
                        System.out.println("Full Response:\n");
                        System.out.println(jsonResponse.toString());

                        // Close the HTTP client.
                        httpClient.close();

                    }
                    case 1 -> {
                        System.out.println("\nChecking the health of load
balancer targets:\n");
                        List<TargetHealthDescription> health =
loadBalancer.checkTargetHealth(targetGroupName);
```

```
                        for (TargetHealthDescription target : health) {
                            System.out.printf("\tTarget %s on port %d is %s%n",
 target.target().id(),
                                    target.target().port(),
 target.targetHealth().stateAsString());
                        }
                        System.out.println("""
                                Note that it can take a minute or two for the
 health check to update
                                after changes are made.
                                """);
                    }
                    case 2 -> {
                        System.out.println("\nOkay, let's move on.");
                        System.out.println("-".repeat(88));
                        return; // Exit the method when choice is 2
                    }
                    default -> System.out.println("You must choose a value
 between 0-2. Please select again.");
                }

            } catch (java.util.InputMismatchException e) {
                System.out.println("Invalid input. Please select again.");
                scanner.nextLine(); // Clear the input buffer.
            }
        }
    }

    public static String readFileAsString(String filePath) throws IOException {
        byte[] bytes = Files.readAllBytes(Paths.get(filePath));
        return new String(bytes);
    }
}
```

Créez une classe qui englobe les actions Auto Scaling et Amazon EC2.

```
public class AutoScaler {

    private static Ec2Client ec2Client;
    private static AutoScalingClient autoScalingClient;
    private static IamClient iamClient;
```

```
    private static SsmClient ssmClient;

    private IamClient getIAMClient() {
        if (iamClient == null) {
            iamClient = IamClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return iamClient;
    }

    private SsmClient getSSMClient() {
        if (ssmClient == null) {
            ssmClient = SsmClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ssmClient;
    }

    private Ec2Client getEc2Client() {
        if (ec2Client == null) {
            ec2Client = Ec2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ec2Client;
    }

    private AutoScalingClient getAutoScalingClient() {
        if (autoScalingClient == null) {
            autoScalingClient = AutoScalingClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return autoScalingClient;
    }

    /**
     * Terminates and instances in an EC2 Auto Scaling group. After an instance
is
     * terminated, it can no longer be accessed.
     */
    public void terminateInstance(String instanceId) {
```

```
        TerminateInstanceInAutoScalingGroupRequest terminateInstanceIRequest =
TerminateInstanceInAutoScalingGroupRequest
                .builder()
                .instanceId(instanceId)
                .shouldDecrementDesiredCapacity(false)
                .build();


getAutoScalingClient().terminateInstanceInAutoScalingGroup(terminateInstanceIRequest);
        System.out.format("Terminated instance %s.", instanceId);
    }

    /**
     * Replaces the profile associated with a running instance. After the profile
is
     * replaced, the instance is rebooted to ensure that it uses the new profile.
     * When
     * the instance is ready, Systems Manager is used to restart the Python web
     * server.
     */
    public void replaceInstanceProfile(String instanceId, String
newInstanceProfileName, String profileAssociationId)
            throws InterruptedException {
        // Create an IAM instance profile specification.
        software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
iamInstanceProfile =
software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
                .builder()
                .name(newInstanceProfileName) // Make sure
'newInstanceProfileName' is a valid IAM Instance Profile
                                              // name.
                .build();

        // Replace the IAM instance profile association for the EC2 instance.
        ReplaceIamInstanceProfileAssociationRequest replaceRequest =
ReplaceIamInstanceProfileAssociationRequest
                .builder()
                .iamInstanceProfile(iamInstanceProfile)
                .associationId(profileAssociationId) // Make sure
'profileAssociationId' is a valid association ID.
                .build();

        try {
            getEc2Client().replaceIamInstanceProfileAssociation(replaceRequest);
```

```
            // Handle the response as needed.
        } catch (Ec2Exception e) {
            // Handle exceptions, log, or report the error.
            System.err.println("Error: " + e.getMessage());
        }
        System.out.format("Replaced instance profile for association %s with
profile %s.", profileAssociationId,
                newInstanceProfileName);
        TimeUnit.SECONDS.sleep(15);
        boolean instReady = false;
        int tries = 0;

        // Reboot after 60 seconds
        while (!instReady) {
            if (tries % 6 == 0) {
                getEc2Client().rebootInstances(RebootInstancesRequest.builder()
                        .instanceIds(instanceId)
                        .build());
                System.out.println("Rebooting instance " + instanceId + " and
waiting for it to be ready.");
            }
            tries++;
            try {
                TimeUnit.SECONDS.sleep(10);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }

            DescribeInstanceInformationResponse informationResponse =
getSSMClient().describeInstanceInformation();
            List<InstanceInformation> instanceInformationList =
informationResponse.instanceInformationList();
            for (InstanceInformation info : instanceInformationList) {
                if (info.instanceId().equals(instanceId)) {
                    instReady = true;
                    break;
                }
            }
        }

        SendCommandRequest sendCommandRequest = SendCommandRequest.builder()
                .instanceIds(instanceId)
                .documentName("AWS-RunShellScript")
                .parameters(Collections.singletonMap("commands",
```

```
                        Collections.singletonList("cd / && sudo python3 server.py
80")))
                .build();

        getSSMClient().sendCommand(sendCommandRequest);
        System.out.println("Restarted the Python web server on instance " +
instanceId + ".");
    }

    public void openInboundPort(String secGroupId, String port, String ipAddress)
{
        AuthorizeSecurityGroupIngressRequest ingressRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                .groupName(secGroupId)
                .cidrIp(ipAddress)
                .fromPort(Integer.parseInt(port))
                .build();

        getEc2Client().authorizeSecurityGroupIngress(ingressRequest);
        System.out.format("Authorized ingress to %s on port %s from %s.",
secGroupId, port, ipAddress);
    }

    /**
     * Detaches a role from an instance profile, detaches policies from the role,
     * and deletes all the resources.
     */
    public void deleteInstanceProfile(String roleName, String profileName) {
        try {
            software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
getInstanceProfileRequest =
software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
                    .builder()
                    .instanceProfileName(profileName)
                    .build();

            GetInstanceProfileResponse response =
getIAMClient().getInstanceProfile(getInstanceProfileRequest);
            String name = response.instanceProfile().instanceProfileName();
            System.out.println(name);

            RemoveRoleFromInstanceProfileRequest profileRequest =
RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
```

```
                    .roleName(roleName)
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(profileRequest);
            DeleteInstanceProfileRequest deleteInstanceProfileRequest =
DeleteInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .build();

            getIAMClient().deleteInstanceProfile(deleteInstanceProfileRequest);
            System.out.println("Deleted instance profile " + profileName);

            DeleteRoleRequest deleteRoleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();

            // List attached role policies.
            ListAttachedRolePoliciesResponse rolesResponse = getIAMClient()
                    .listAttachedRolePolicies(role -> role.roleName(roleName));
            List<AttachedPolicy> attachedPolicies =
rolesResponse.attachedPolicies();
            for (AttachedPolicy attachedPolicy : attachedPolicies) {
                DetachRolePolicyRequest request =
DetachRolePolicyRequest.builder()
                        .roleName(roleName)
                        .policyArn(attachedPolicy.policyArn())
                        .build();

                getIAMClient().detachRolePolicy(request);
                System.out.println("Detached and deleted policy " +
attachedPolicy.policyName());
            }

            getIAMClient().deleteRole(deleteRoleRequest);
            System.out.println("Instance profile and role deleted.");

        } catch (IamException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public void deleteTemplate(String templateName) {
```

```
        getEc2Client().deleteLaunchTemplate(name ->
name.launchTemplateName(templateName));
        System.out.format(templateName + " was deleted.");
    }

    public void deleteAutoScaleGroup(String groupName) {
        DeleteAutoScalingGroupRequest deleteAutoScalingGroupRequest =
DeleteAutoScalingGroupRequest.builder()
                .autoScalingGroupName(groupName)
                .forceDelete(true)
                .build();


getAutoScalingClient().deleteAutoScalingGroup(deleteAutoScalingGroupRequest);
        System.out.println(groupName + " was deleted.");
    }

    /*
     * Verify the default security group of the specified VPC allows ingress from
     * this
     * computer. This can be done by allowing ingress from this computer's IP
     * address. In some situations, such as connecting from a corporate network,
you
     * must instead specify a prefix list ID. You can also temporarily open the
port
     * to
     * any IP address while running this example. If you do, be sure to remove
     * public
     * access when you're done.
     *
     */
    public GroupInfo verifyInboundPort(String VPC, int port, String ipAddress) {
        boolean portIsOpen = false;
        GroupInfo groupInfo = new GroupInfo();
        try {
            Filter filter = Filter.builder()
                    .name("group-name")
                    .values("default")
                    .build();

            Filter filter1 = Filter.builder()
                    .name("vpc-id")
                    .values(VPC)
                    .build();
```

```
            DescribeSecurityGroupsRequest securityGroupsRequest =
DescribeSecurityGroupsRequest.builder()
                    .filters(filter, filter1)
                    .build();

            DescribeSecurityGroupsResponse securityGroupsResponse =
getEc2Client()
                    .describeSecurityGroups(securityGroupsRequest);
            String securityGroup =
securityGroupsResponse.securityGroups().get(0).groupName();
            groupInfo.setGroupName(securityGroup);

            for (SecurityGroup secGroup :
securityGroupsResponse.securityGroups()) {
                System.out.println("Found security group: " +
secGroup.groupId());

                for (IpPermission ipPermission : secGroup.ipPermissions()) {
                    if (ipPermission.fromPort() == port) {
                        System.out.println("Found inbound rule: " +
ipPermission);
                        for (IpRange ipRange : ipPermission.ipRanges()) {
                            String cidrIp = ipRange.cidrIp();
                            if (cidrIp.startsWith(ipAddress) ||
cidrIp.equals("0.0.0.0/0")) {
                                System.out.println(cidrIp + " is applicable");
                                portIsOpen = true;
                            }
                        }

                        if (!ipPermission.prefixListIds().isEmpty()) {
                            System.out.println("Prefix lList is applicable");
                            portIsOpen = true;
                        }

                        if (!portIsOpen) {
                            System.out
                                    .println("The inbound rule does not appear to
be open to either this computer's IP,"
                                            + " all IP addresses (0.0.0.0/0), or
to a prefix list ID.");
                        } else {
                            break;
```

```
                        }
                    }
                }
            }

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }

        groupInfo.setPortOpen(portIsOpen);
        return groupInfo;
    }

    /*
     * Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
     * Scaling group.
     * The target group specifies how the load balancer forward requests to the
     * instances
     * in the group.
     */
    public void attachLoadBalancerTargetGroup(String asGroupName, String
targetGroupARN) {
        try {
            AttachLoadBalancerTargetGroupsRequest targetGroupsRequest =
AttachLoadBalancerTargetGroupsRequest.builder()
                    .autoScalingGroupName(asGroupName)
                    .targetGroupARNs(targetGroupARN)
                    .build();


getAutoScalingClient().attachLoadBalancerTargetGroups(targetGroupsRequest);
            System.out.println("Attached load balancer to " + asGroupName);

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Creates an EC2 Auto Scaling group with the specified size.
    public String[] createGroup(int groupSize, String templateName, String
autoScalingGroupName) {

        // Get availability zones.
```

```
software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
zonesRequest =
software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
               .builder()
               .build();

       DescribeAvailabilityZonesResponse zonesResponse =
getEc2Client().describeAvailabilityZones(zonesRequest);
       List<String> availabilityZoneNames =
zonesResponse.availabilityZones().stream()

.map(software.amazon.awssdk.services.ec2.model.AvailabilityZone::zoneName)
               .collect(Collectors.toList());

       String availabilityZones = String.join(",", availabilityZoneNames);
       LaunchTemplateSpecification specification =
LaunchTemplateSpecification.builder()
               .launchTemplateName(templateName)
               .version("$Default")
               .build();

       String[] zones = availabilityZones.split(",");
       CreateAutoScalingGroupRequest groupRequest =
CreateAutoScalingGroupRequest.builder()
               .launchTemplate(specification)
               .availabilityZones(zones)
               .maxSize(groupSize)
               .minSize(groupSize)
               .autoScalingGroupName(autoScalingGroupName)
               .build();

       try {
           getAutoScalingClient().createAutoScalingGroup(groupRequest);

       } catch (AutoScalingException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
       System.out.println("Created an EC2 Auto Scaling group named " +
autoScalingGroupName);
       return zones;
   }
```

```
   public String getDefaultVPC() {
       // Define the filter.
       Filter defaultFilter = Filter.builder()
               .name("is-default")
               .values("true")
               .build();

       software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest request =
software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest
               .builder()
               .filters(defaultFilter)
               .build();

       DescribeVpcsResponse response = getEc2Client().describeVpcs(request);
       return response.vpcs().get(0).vpcId();
   }

   // Gets the default subnets in a VPC for a specified list of Availability
Zones.
   public List<Subnet> getSubnets(String vpcId, String[] availabilityZones) {
       List<Subnet> subnets = null;
       Filter vpcFilter = Filter.builder()
               .name("vpc-id")
               .values(vpcId)
               .build();

       Filter azFilter = Filter.builder()
               .name("availability-zone")
               .values(availabilityZones)
               .build();

       Filter defaultForAZ = Filter.builder()
               .name("default-for-az")
               .values("true")
               .build();

       DescribeSubnetsRequest request = DescribeSubnetsRequest.builder()
               .filters(vpcFilter, azFilter, defaultForAZ)
               .build();

       DescribeSubnetsResponse response =
getEc2Client().describeSubnets(request);
       subnets = response.subnets();
       return subnets;
```

```
    }

    // Gets data about the instances in the EC2 Auto Scaling group.
    public String getBadInstance(String groupName) {
        DescribeAutoScalingGroupsRequest request =
DescribeAutoScalingGroupsRequest.builder()
                .autoScalingGroupNames(groupName)
                .build();

        DescribeAutoScalingGroupsResponse response =
getAutoScalingClient().describeAutoScalingGroups(request);
        AutoScalingGroup autoScalingGroup = response.autoScalingGroups().get(0);
        List<String> instanceIds = autoScalingGroup.instances().stream()
                .map(instance -> instance.instanceId())
                .collect(Collectors.toList());

        String[] instanceIdArray = instanceIds.toArray(new String[0]);
        for (String instanceId : instanceIdArray) {
            System.out.println("Instance ID: " + instanceId);
            return instanceId;
        }
        return "";
    }

    // Gets data about the profile associated with an instance.
    public String getInstanceProfile(String instanceId) {
        Filter filter = Filter.builder()
                .name("instance-id")
                .values(instanceId)
                .build();

        DescribeIamInstanceProfileAssociationsRequest associationsRequest =
DescribeIamInstanceProfileAssociationsRequest
                .builder()
                .filters(filter)
                .build();

        DescribeIamInstanceProfileAssociationsResponse response = getEc2Client()
                .describeIamInstanceProfileAssociations(associationsRequest);
        return response.iamInstanceProfileAssociations().get(0).associationId();
    }

    public void deleteRolesPolicies(String policyName, String roleName, String
InstanceProfile) {
```

```
        ListPoliciesRequest listPoliciesRequest =
 ListPoliciesRequest.builder().build();
        ListPoliciesResponse listPoliciesResponse =
 getIAMClient().listPolicies(listPoliciesRequest);
        for (Policy policy : listPoliciesResponse.policies()) {
            if (policy.policyName().equals(policyName)) {
                // List the entities (users, groups, roles) that are attached to
 the policy.

 software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
 listEntitiesRequest =
 software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
                        .builder()
                        .policyArn(policy.arn())
                        .build();
                ListEntitiesForPolicyResponse listEntitiesResponse = iamClient
                        .listEntitiesForPolicy(listEntitiesRequest);
                if (!listEntitiesResponse.policyGroups().isEmpty() || !
listEntitiesResponse.policyUsers().isEmpty()
                        || !listEntitiesResponse.policyRoles().isEmpty()) {
                    // Detach the policy from any entities it is attached to.
                    DetachRolePolicyRequest detachPolicyRequest =
 DetachRolePolicyRequest.builder()
                            .policyArn(policy.arn())
                            .roleName(roleName) // Specify the name of the IAM
 role
                            .build();

                    getIAMClient().detachRolePolicy(detachPolicyRequest);
                    System.out.println("Policy detached from entities.");
                }

                // Now, you can delete the policy.
                DeletePolicyRequest deletePolicyRequest =
 DeletePolicyRequest.builder()
                        .policyArn(policy.arn())
                        .build();

                getIAMClient().deletePolicy(deletePolicyRequest);
                System.out.println("Policy deleted successfully.");
                break;
            }
        }
```

```
        // List the roles associated with the instance profile
        ListInstanceProfilesForRoleRequest listRolesRequest =
 ListInstanceProfilesForRoleRequest.builder()
                .roleName(roleName)
                .build();

        // Detach the roles from the instance profile
        ListInstanceProfilesForRoleResponse listRolesResponse =
 iamClient.listInstanceProfilesForRole(listRolesRequest);
        for (software.amazon.awssdk.services.iam.model.InstanceProfile profile :
 listRolesResponse.instanceProfiles()) {
            RemoveRoleFromInstanceProfileRequest removeRoleRequest =
 RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(InstanceProfile)
                    .roleName(roleName) // Remove the extra dot here
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(removeRoleRequest);
            System.out.println("Role " + roleName + " removed from instance
 profile " + InstanceProfile);
        }

        // Delete the instance profile after removing all roles
        DeleteInstanceProfileRequest deleteInstanceProfileRequest =
 DeleteInstanceProfileRequest.builder()
                .instanceProfileName(InstanceProfile)
                .build();

        getIAMClient().deleteInstanceProfile(r ->
 r.instanceProfileName(InstanceProfile));
        System.out.println(InstanceProfile + " Deleted");
        System.out.println("All roles and policies are deleted.");
    }
}
```

Créez une classe qui englobe les actions Elastic Load Balancing.

```
public class LoadBalancer {
    public ElasticLoadBalancingV2Client elasticLoadBalancingV2Client;

    public ElasticLoadBalancingV2Client getLoadBalancerClient() {
        if (elasticLoadBalancingV2Client == null) {
```

```
            elasticLoadBalancingV2Client = ElasticLoadBalancingV2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }

        return elasticLoadBalancingV2Client;
    }

    // Checks the health of the instances in the target group.
    public List<TargetHealthDescription> checkTargetHealth(String
targetGroupName) {
        DescribeTargetGroupsRequest targetGroupsRequest =
DescribeTargetGroupsRequest.builder()
                .names(targetGroupName)
                .build();

        DescribeTargetGroupsResponse tgResponse =
getLoadBalancerClient().describeTargetGroups(targetGroupsRequest);

        DescribeTargetHealthRequest healthRequest =
DescribeTargetHealthRequest.builder()

.targetGroupArn(tgResponse.targetGroups().get(0).targetGroupArn())
                .build();

        DescribeTargetHealthResponse healthResponse =
getLoadBalancerClient().describeTargetHealth(healthRequest);
        return healthResponse.targetHealthDescriptions();
    }

    // Gets the HTTP endpoint of the load balancer.
    public String getEndpoint(String lbName) {
        DescribeLoadBalancersResponse res = getLoadBalancerClient()
                .describeLoadBalancers(describe -> describe.names(lbName));
        return res.loadBalancers().get(0).dnsName();
    }

    // Deletes a load balancer.
    public void deleteLoadBalancer(String lbName) {
        try {
            // Use a waiter to delete the Load Balancer.
            DescribeLoadBalancersResponse res = getLoadBalancerClient()
                    .describeLoadBalancers(describe -> describe.names(lbName));
```

```
            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()

.loadBalancerArns(res.loadBalancers().get(0).loadBalancerArn())
                    .build();

            getLoadBalancerClient().deleteLoadBalancer(
                    builder ->
builder.loadBalancerArn(res.loadBalancers().get(0).loadBalancerArn()));
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                    .waitUntilLoadBalancersDeleted(request);
            waiterResponse.matched().response().ifPresent(System.out::println);

        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(lbName + " was deleted.");
    }

    // Deletes the target group.
    public void deleteTargetGroup(String targetGroupName) {
        try {
            DescribeTargetGroupsResponse res = getLoadBalancerClient()
                    .describeTargetGroups(describe ->
describe.names(targetGroupName));
            getLoadBalancerClient()
                    .deleteTargetGroup(builder ->
builder.targetGroupArn(res.targetGroups().get(0).targetGroupArn()));
        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(targetGroupName + " was deleted.");
    }

    // Verify this computer can successfully send a GET request to the load
balancer
    // endpoint.
    public boolean verifyLoadBalancerEndpoint(String elbDnsName) throws
IOException, InterruptedException {
        boolean success = false;
        int retries = 3;
```

```
        CloseableHttpClient httpClient = HttpClients.createDefault();

        // Create an HTTP GET request to the ELB.
        HttpGet httpGet = new HttpGet("http://" + elbDnsName);
        try {
            while ((!success) && (retries > 0)) {
                // Execute the request and get the response.
                HttpResponse response = httpClient.execute(httpGet);
                int statusCode = response.getStatusLine().getStatusCode();
                System.out.println("HTTP Status Code: " + statusCode);
                if (statusCode == 200) {
                    success = true;
                } else {
                    retries--;
                    System.out.println("Got connection error from load balancer
endpoint, retrying...");
                    TimeUnit.SECONDS.sleep(15);
                }
            }

        } catch (org.apache.http.conn.HttpHostConnectException e) {
            System.out.println(e.getMessage());
        }

        System.out.println("Status.." + success);
        return success;
    }

    /*
     * Creates an Elastic Load Balancing target group. The target group specifies
     * how
     * the load balancer forward requests to instances in the group and how
instance
     * health is checked.
     */
    public String createTargetGroup(String protocol, int port, String vpcId,
String targetGroupName) {
        CreateTargetGroupRequest targetGroupRequest =
CreateTargetGroupRequest.builder()
                .healthCheckPath("/healthcheck")
                .healthCheckTimeoutSeconds(5)
                .port(port)
                .vpcId(vpcId)
                .name(targetGroupName)
```

```
                .protocol(protocol)
                .build();

        CreateTargetGroupResponse targetGroupResponse =
getLoadBalancerClient().createTargetGroup(targetGroupRequest);
        String targetGroupArn =
targetGroupResponse.targetGroups().get(0).targetGroupArn();
        String targetGroup =
targetGroupResponse.targetGroups().get(0).targetGroupName();
        System.out.println("The " + targetGroup + " was created with ARN" +
targetGroupArn);
        return targetGroupArn;
    }

    /*
     * Creates an Elastic Load Balancing load balancer that uses the specified
     * subnets
     * and forwards requests to the specified target group.
     */
    public String createLoadBalancer(List<Subnet> subnetIds, String
targetGroupARN, String lbName, int port,
            String protocol) {
        try {
            List<String> subnetIdStrings = subnetIds.stream()
                    .map(Subnet::subnetId)
                    .collect(Collectors.toList());

            CreateLoadBalancerRequest balancerRequest =
CreateLoadBalancerRequest.builder()
                    .subnets(subnetIdStrings)
                    .name(lbName)
                    .scheme("internet-facing")
                    .build();

            // Create and wait for the load balancer to become available.
            CreateLoadBalancerResponse lsResponse =
getLoadBalancerClient().createLoadBalancer(balancerRequest);
            String lbARN = lsResponse.loadBalancers().get(0).loadBalancerArn();

            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()
                    .loadBalancerArns(lbARN)
```

```
                    .build();

            System.out.println("Waiting for Load Balancer " + lbName + " to
 become available.");
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
 loadBalancerWaiter
                    .waitUntilLoadBalancerAvailable(request);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Load Balancer " + lbName + " is available.");

            // Get the DNS name (endpoint) of the load balancer.
            String lbDNSName = lsResponse.loadBalancers().get(0).dnsName();
            System.out.println("*** Load Balancer DNS Name: " + lbDNSName);

            // Create a listener for the load balance.
            Action action = Action.builder()
                    .targetGroupArn(targetGroupARN)
                    .type("forward")
                    .build();

            CreateListenerRequest listenerRequest =
 CreateListenerRequest.builder()

 .loadBalancerArn(lsResponse.loadBalancers().get(0).loadBalancerArn())
                    .defaultActions(action)
                    .port(port)
                    .protocol(protocol)
                    .defaultActions(action)
                    .build();

            getLoadBalancerClient().createListener(listenerRequest);
            System.out.println("Created listener to forward traffic from load
 balancer " + lbName + " to target group "
                    + targetGroupARN);

            // Return the load balancer DNS name.
            return lbDNSName;

        } catch (ElasticLoadBalancingV2Exception e) {
            e.printStackTrace();
        }
        return "";
    }
}
```

Créez une classe qui utilise DynamoDB pour simuler un service de recommandation.

```
public class Database {

    private static DynamoDbClient dynamoDbClient;

    public static DynamoDbClient getDynamoDbClient() {
        if (dynamoDbClient == null) {
            dynamoDbClient = DynamoDbClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return dynamoDbClient;
    }

    // Checks to see if the Amazon DynamoDB table exists.
    private boolean doesTableExist(String tableName) {
        try {
            // Describe the table and catch any exceptions.
            DescribeTableRequest describeTableRequest =
 DescribeTableRequest.builder()
                    .tableName(tableName)
                    .build();

            getDynamoDbClient().describeTable(describeTableRequest);
            System.out.println("Table '" + tableName + "' exists.");
            return true;

        } catch (ResourceNotFoundException e) {
            System.out.println("Table '" + tableName + "' does not exist.");
        } catch (DynamoDbException e) {
            System.err.println("Error checking table existence: " +
 e.getMessage());
        }
        return false;
    }

    /*
     * Creates a DynamoDB table to use a recommendation service. The table has a
     * hash key named 'MediaType' that defines the type of media recommended,
 such
```

```
     * as
     * Book or Movie, and a range key named 'ItemId' that, combined with the
     * MediaType,
     * forms a unique identifier for the recommended item.
     */
    public void createTable(String tableName, String fileName) throws IOException
{
        // First check to see if the table exists.
        boolean doesExist = doesTableExist(tableName);
        if (!doesExist) {
            DynamoDbWaiter dbWaiter = getDynamoDbClient().waiter();
            CreateTableRequest createTableRequest = CreateTableRequest.builder()
                    .tableName(tableName)
                    .attributeDefinitions(
                            AttributeDefinition.builder()
                                    .attributeName("MediaType")
                                    .attributeType(ScalarAttributeType.S)
                                    .build(),
                            AttributeDefinition.builder()
                                    .attributeName("ItemId")
                                    .attributeType(ScalarAttributeType.N)
                                    .build())
                    .keySchema(
                            KeySchemaElement.builder()
                                    .attributeName("MediaType")
                                    .keyType(KeyType.HASH)
                                    .build(),
                            KeySchemaElement.builder()
                                    .attributeName("ItemId")
                                    .keyType(KeyType.RANGE)
                                    .build())
                    .provisionedThroughput(
                            ProvisionedThroughput.builder()
                                    .readCapacityUnits(5L)
                                    .writeCapacityUnits(5L)
                                    .build())
                    .build();

            getDynamoDbClient().createTable(createTableRequest);
            System.out.println("Creating table " + tableName + "...");

            // Wait until the Amazon DynamoDB table is created.
            DescribeTableRequest tableRequest = DescribeTableRequest.builder()
                    .tableName(tableName)
```

```
                    .build();

            WaiterResponse<DescribeTableResponse> waiterResponse =
dbWaiter.waitUntilTableExists(tableRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Table " + tableName + " created.");

            // Add records to the table.
            populateTable(fileName, tableName);
        }
    }

    public void deleteTable(String tableName) {
        getDynamoDbClient().deleteTable(table -> table.tableName(tableName));
        System.out.println("Table " + tableName + " deleted.");
    }

    // Populates the table with data located in a JSON file using the DynamoDB
    // enhanced client.
    public void populateTable(String fileName, String tableName) throws
IOException {
        DynamoDbEnhancedClient enhancedClient = DynamoDbEnhancedClient.builder()
                .dynamoDbClient(getDynamoDbClient())
                .build();
        ObjectMapper objectMapper = new ObjectMapper();
        File jsonFile = new File(fileName);
        JsonNode rootNode = objectMapper.readTree(jsonFile);

        DynamoDbTable<Recommendation> mappedTable =
enhancedClient.table(tableName,
                TableSchema.fromBean(Recommendation.class));
        for (JsonNode currentNode : rootNode) {
            String mediaType = currentNode.path("MediaType").path("S").asText();
            int itemId = currentNode.path("ItemId").path("N").asInt();
            String title = currentNode.path("Title").path("S").asText();
            String creator = currentNode.path("Creator").path("S").asText();

            // Create a Recommendation object and set its properties.
            Recommendation rec = new Recommendation();
            rec.setMediaType(mediaType);
            rec.setItemId(itemId);
            rec.setTitle(title);
            rec.setCreator(creator);
```

```
            // Put the item into the DynamoDB table.
            mappedTable.putItem(rec); // Add the Recommendation to the list.
        }
        System.out.println("Added all records to the " + tableName);
    }
}
```

Créez une classe qui englobe les actions de Systems Manager.

```
public class ParameterHelper {

    String tableName = "doc-example-resilient-architecture-table";
    String dyntable = "doc-example-recommendation-service";
    String failureResponse = "doc-example-resilient-architecture-failure-
response";
    String healthCheck = "doc-example-resilient-architecture-health-check";

    public void reset() {
        put(dyntable, tableName);
        put(failureResponse, "none");
        put(healthCheck, "shallow");
    }

    public void put(String name, String value) {
        SsmClient ssmClient = SsmClient.builder()
                .region(Region.US_EAST_1)
                .build();

        PutParameterRequest parameterRequest = PutParameterRequest.builder()
                .name(name)
                .value(value)
                .overwrite(true)
                .type("String")
                .build();

        ssmClient.putParameter(parameterRequest);
        System.out.printf("Setting demo parameter %s to '%s'.", name, value);
    }
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK for Java 2.x .
  - AttachLoadBalancerTargetGroupes
  - CreateAutoScalingGroup
  - CreateInstanceProfil
  - CreateLaunchModèle
  - CreateListener
  - CreateLoadÉquilibreur
  - CreateTargetGroupe
  - DeleteAutoScalingGroup
  - DeleteInstanceProfil
  - DeleteLaunchModèle
  - DeleteLoadÉquilibreur
  - DeleteTargetGroupe
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadÉquilibreurs
  - DescribeSubnets
  - DescribeTargetGroupes
  - DescribeTargetHealth
  - DescribeVpcs
  - RebootInstances
  - ReplaceIamInstanceProfileAssociation
  - TerminateInstanceInAutoScalingGroup
  - UpdateAutoScalingGroup

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Exécutez un scénario interactif à une invite de commande.

```
#!/usr/bin/env node
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0


import {
  Scenario,
  parseScenarioArgs,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";

/**
 * The workflow steps are split into three stages:
 *   - deploy
 *   - demo
 *   - destroy
 *
 * Each of these stages has a corresponding file prefixed with steps-*.
 */
import { deploySteps } from "./steps-deploy.js";
import { demoSteps } from "./steps-demo.js";
import { destroySteps } from "./steps-destroy.js";

/**
 * The context is passed to every scenario. Scenario steps
 * will modify the context.
 */
const context = {};

/**
 * Three Scenarios are created for the workflow. A Scenario is an orchestration
 class
```

```
 * that simplifies running a series of steps.
 */
export const scenarios = {
  // Deploys all resources necessary for the workflow.
  deploy: new Scenario("Resilient Workflow - Deploy", deploySteps, context),
  // Demonstrates how a fragile web service can be made more resilient.
  demo: new Scenario("Resilient Workflow - Demo", demoSteps, context),
  // Destroys the resources created for the workflow.
  destroy: new Scenario("Resilient Workflow - Destroy", destroySteps, context),
};

// Call function if run directly
import { fileURLToPath } from "url";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios);
}
```

Créez des étapes pour déployer toutes les ressources.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { join } from "node:path";
import { readFileSync, writeFileSync } from "node:fs";
import axios from "axios";

import {
  BatchWriteItemCommand,
  CreateTableCommand,
  DynamoDBClient,
  waitUntilTableExists,
} from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  CreateKeyPairCommand,
  CreateLaunchTemplateCommand,
  DescribeAvailabilityZonesCommand,
  DescribeVpcsCommand,
  DescribeSubnetsCommand,
  DescribeSecurityGroupsCommand,
  AuthorizeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
```

```
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  AttachRolePolicyCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import { SSMClient, GetParameterCommand } from "@aws-sdk/client-ssm";
import {
  CreateAutoScalingGroupCommand,
  AutoScalingClient,
  AttachLoadBalancerTargetGroupsCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  CreateListenerCommand,
  CreateLoadBalancerCommand,
  CreateTargetGroupCommand,
  ElasticLoadBalancingV2Client,
  waitUntilLoadBalancerAvailable,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH, ROOT } from "./constants.js";
import { initParamsSteps } from "./steps-reset-params.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const deploySteps = [
  new ScenarioOutput("introduction", MESSAGES.introduction, { header: true }),
  new ScenarioInput("confirmDeployment", MESSAGES.confirmDeployment, {
    type: "confirm",
  }),
  new ScenarioAction(
    "handleConfirmDeployment",
    (c) => c.confirmDeployment === false && process.exit(),
```

```
  ),
  new ScenarioOutput(
    "creatingTable",
    MESSAGES.creatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("createTable", async () => {
    const client = new DynamoDBClient({});
    await client.send(
      new CreateTableCommand({
        TableName: NAMES.tableName,
        ProvisionedThroughput: {
          ReadCapacityUnits: 5,
          WriteCapacityUnits: 5,
        },
        AttributeDefinitions: [
          {
            AttributeName: "MediaType",
            AttributeType: "S",
          },
          {
            AttributeName: "ItemId",
            AttributeType: "N",
          },
        ],
        KeySchema: [
          {
            AttributeName: "MediaType",
            KeyType: "HASH",
          },
          {
            AttributeName: "ItemId",
            KeyType: "RANGE",
          },
        ],
      }),
    );
    await waitUntilTableExists({ client }, { TableName: NAMES.tableName });
  }),
  new ScenarioOutput(
    "createdTable",
    MESSAGES.createdTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "populatingTable",
```

```
    MESSAGES.populatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("populateTable", () => {
    const client = new DynamoDBClient({});
    /**
     * @type {{ default: import("@aws-sdk/client-dynamodb").PutRequest['Item']
[] }}
     */
    const recommendations = JSON.parse(
      readFileSync(join(RESOURCES_PATH, "recommendations.json")),
    );

    return client.send(
      new BatchWriteItemCommand({
        RequestItems: {
          [NAMES.tableName]: recommendations.map((item) => ({
            PutRequest: { Item: item },
          })),
        },
      }),
    );
  }),
  new ScenarioOutput(
    "populatedTable",
    MESSAGES.populatedTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "creatingKeyPair",
    MESSAGES.creatingKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioAction("createKeyPair", async () => {
    const client = new EC2Client({});
    const { KeyMaterial } = await client.send(
      new CreateKeyPairCommand({
        KeyName: NAMES.keyPairName,
      }),
    );

    writeFileSync(`${NAMES.keyPairName}.pem`, KeyMaterial, { mode: 0o600 });
  }),
  new ScenarioOutput(
    "createdKeyPair",
    MESSAGES.createdKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
```

```
  new ScenarioOutput(
    "creatingInstancePolicy",
    MESSAGES.creatingInstancePolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    ),
  ),
  new ScenarioAction("createInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const {
      Policy: { Arn },
    } = await client.send(
      new CreatePolicyCommand({
        PolicyName: NAMES.instancePolicyName,
        PolicyDocument: readFileSync(
          join(RESOURCES_PATH, "instance_policy.json"),
        ),
      }),
    );
    state.instancePolicyArn = Arn;
  }),
  new ScenarioOutput("createdInstancePolicy", (state) =>
    MESSAGES.createdInstancePolicy
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_POLICY_ARN}", state.instancePolicyArn),
  ),
  new ScenarioOutput(
    "creatingInstanceRole",
    MESSAGES.creatingInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioAction("createInstanceRole", () => {
    const client = new IAMClient({});
    return client.send(
      new CreateRoleCommand({
        RoleName: NAMES.instanceRoleName,
        AssumeRolePolicyDocument: readFileSync(
          join(ROOT, "assume-role-policy.json"),
        ),
      }),
    );
  }),
```

```
  new ScenarioOutput(
    "createdInstanceRole",
    MESSAGES.createdInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioOutput(
    "attachingPolicyToRole",
    MESSAGES.attachingPolicyToRole
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName)
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName),
  ),
  new ScenarioAction("attachPolicyToRole", async (state) => {
    const client = new IAMClient({});
    await client.send(
      new AttachRolePolicyCommand({
        RoleName: NAMES.instanceRoleName,
        PolicyArn: state.instancePolicyArn,
      }),
    );
  }),
  new ScenarioOutput(
    "attachedPolicyToRole",
    MESSAGES.attachedPolicyToRole
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioOutput(
    "creatingInstanceProfile",
    MESSAGES.creatingInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    ),
  ),
  new ScenarioAction("createInstanceProfile", async (state) => {
    const client = new IAMClient({});
    const {
      InstanceProfile: { Arn },
    } = await client.send(
      new CreateInstanceProfileCommand({
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
```

```
    state.instanceProfileArn = Arn;

    await waitUntilInstanceProfileExists(
      { client },
      { InstanceProfileName: NAMES.instanceProfileName },
    );
  }),
  new ScenarioOutput("createdInstanceProfile", (state) =>
    MESSAGES.createdInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_PROFILE_ARN}", state.instanceProfileArn),
  ),
  new ScenarioOutput(
    "addingRoleToInstanceProfile",
    MESSAGES.addingRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioAction("addRoleToInstanceProfile", () => {
    const client = new IAMClient({});
    return client.send(
      new AddRoleToInstanceProfileCommand({
        RoleName: NAMES.instanceRoleName,
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
  }),
  new ScenarioOutput(
    "addedRoleToInstanceProfile",
    MESSAGES.addedRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  ...initParamsSteps,
  new ScenarioOutput("creatingLaunchTemplate", MESSAGES.creatingLaunchTemplate),
  new ScenarioAction("createLaunchTemplate", async () => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
    const ssmClient = new SSMClient({});
    const { Parameter } = await ssmClient.send(
      new GetParameterCommand({
        Name: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
      }),
    );
    const ec2Client = new EC2Client({});
```

```
    await ec2Client.send(
      new CreateLaunchTemplateCommand({
        LaunchTemplateName: NAMES.launchTemplateName,
        LaunchTemplateData: {
          InstanceType: "t3.micro",
          ImageId: Parameter.Value,
          IamInstanceProfile: { Name: NAMES.instanceProfileName },
          UserData: readFileSync(
            join(RESOURCES_PATH, "server_startup_script.sh"),
          ).toString("base64"),
          KeyName: NAMES.keyPairName,
        },
      }),
      // snippet-end:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
    );
  }),
  new ScenarioOutput(
    "createdLaunchTemplate",
    MESSAGES.createdLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    ),
  ),
  new ScenarioOutput(
    "creatingAutoScalingGroup",
    MESSAGES.creatingAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    ),
  ),
  new ScenarioAction("createAutoScalingGroup", async (state) => {
    const ec2Client = new EC2Client({});
    const { AvailabilityZones } = await ec2Client.send(
      new DescribeAvailabilityZonesCommand({}),
    );
    state.availabilityZoneNames = AvailabilityZones.map((az) => az.ZoneName);
    const autoScalingClient = new AutoScalingClient({});
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new CreateAutoScalingGroupCommand({
          AvailabilityZones: state.availabilityZoneNames,
          AutoScalingGroupName: NAMES.autoScalingGroupName,
          LaunchTemplate: {
            LaunchTemplateName: NAMES.launchTemplateName,
```

```
          Version: "$Default",
        },
        MinSize: 3,
        MaxSize: 3,
      }),
    ),
  );
}),
new ScenarioOutput(
  "createdAutoScalingGroup",
  /**
   * @param {{ availabilityZoneNames: string[] }} state
   */
  (state) =>
    MESSAGES.createdAutoScalingGroup
      .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName)
      .replace(
        "${AVAILABILITY_ZONE_NAMES}",
        state.availabilityZoneNames.join(", "),
      ),
),
new ScenarioInput("confirmContinue", MESSAGES.confirmContinue, {
  type: "confirm",
}),
new ScenarioOutput("loadBalancer", MESSAGES.loadBalancer),
new ScenarioOutput("gettingVpc", MESSAGES.gettingVpc),
new ScenarioAction("getVpc", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DescribeVpcs]
  const client = new EC2Client({});
  const { Vpcs } = await client.send(
    new DescribeVpcsCommand({
      Filters: [{ Name: "is-default", Values: ["true"] }],
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeVpcs]
  state.defaultVpc = Vpcs[0].VpcId;
}),
new ScenarioOutput("gotVpc", (state) =>
  MESSAGES.gotVpc.replace("${VPC_ID}", state.defaultVpc),
),
new ScenarioOutput("gettingSubnets", MESSAGES.gettingSubnets),
new ScenarioAction("getSubnets", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DescribeSubnets]
  const client = new EC2Client({});
```

```
    const { Subnets } = await client.send(
      new DescribeSubnetsCommand({
        Filters: [
          { Name: "vpc-id", Values: [state.defaultVpc] },
          { Name: "availability-zone", Values: state.availabilityZoneNames },
          { Name: "default-for-az", Values: ["true"] },
        ],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.DescribeSubnets]
    state.subnets = Subnets.map((subnet) => subnet.SubnetId);
  }),
  new ScenarioOutput(
    "gotSubnets",
    /**
     * @param {{ subnets: string[] }} state
     */
    (state) =>
      MESSAGES.gotSubnets.replace("${SUBNETS}", state.subnets.join(", ")),
  ),
  new ScenarioOutput(
    "creatingLoadBalancerTargetGroup",
    MESSAGES.creatingLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioAction("createLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateTargetGroup]
    const client = new ElasticLoadBalancingV2Client({});
    const { TargetGroups } = await client.send(
      new CreateTargetGroupCommand({
        Name: NAMES.loadBalancerTargetGroupName,
        Protocol: "HTTP",
        Port: 80,
        HealthCheckPath: "/healthcheck",
        HealthCheckIntervalSeconds: 10,
        HealthCheckTimeoutSeconds: 5,
        HealthyThresholdCount: 2,
        UnhealthyThresholdCount: 2,
        VpcId: state.defaultVpc,
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateTargetGroup]
```

```
    const targetGroup = TargetGroups[0];
    state.targetGroupArn = targetGroup.TargetGroupArn;
    state.targetGroupProtocol = targetGroup.Protocol;
    state.targetGroupPort = targetGroup.Port;
  }),
  new ScenarioOutput(
    "createdLoadBalancerTargetGroup",
    MESSAGES.createdLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioOutput(
    "creatingLoadBalancer",
    MESSAGES.creatingLoadBalancer.replace("${LB_NAME}", NAMES.loadBalancerName),
  ),
  new ScenarioAction("createLoadBalancer", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
    const client = new ElasticLoadBalancingV2Client({});
    const { LoadBalancers } = await client.send(
      new CreateLoadBalancerCommand({
        Name: NAMES.loadBalancerName,
        Subnets: state.subnets,
      }),
    );
    state.loadBalancerDns = LoadBalancers[0].DNSName;
    state.loadBalancerArn = LoadBalancers[0].LoadBalancerArn;
    await waitUntilLoadBalancerAvailable(
      { client },
      { Names: [NAMES.loadBalancerName] },
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
  }),
  new ScenarioOutput("createdLoadBalancer", (state) =>
    MESSAGES.createdLoadBalancer
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioOutput(
    "creatingListener",
    MESSAGES.creatingLoadBalancerListener
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName),
  ),
```

```
  new ScenarioAction("createListener", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateListener]
    const client = new ElasticLoadBalancingV2Client({});
    const { Listeners } = await client.send(
      new CreateListenerCommand({
        LoadBalancerArn: state.loadBalancerArn,
        Protocol: state.targetGroupProtocol,
        Port: state.targetGroupPort,
        DefaultActions: [
          { Type: "forward", TargetGroupArn: state.targetGroupArn },
        ],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateListener]
    const listener = Listeners[0];
    state.loadBalancerListenerArn = listener.ListenerArn;
  }),
  new ScenarioOutput("createdListener", (state) =>
    MESSAGES.createdLoadBalancerListener.replace(
      "${LB_LISTENER_ARN}",
      state.loadBalancerListenerArn,
    ),
  ),
  new ScenarioOutput(
    "attachingLoadBalancerTargetGroup",
    MESSAGES.attachingLoadBalancerTargetGroup
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName)
      .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName),
  ),
  new ScenarioAction("attachLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.AttachTargetGroup]
    const client = new AutoScalingClient({});
    await client.send(
      new AttachLoadBalancerTargetGroupsCommand({
        AutoScalingGroupName: NAMES.autoScalingGroupName,
        TargetGroupARNs: [state.targetGroupArn],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.AttachTargetGroup]
  }),
  new ScenarioOutput(
    "attachedLoadBalancerTargetGroup",
    MESSAGES.attachedLoadBalancerTargetGroup,
  ),
```

```
  new ScenarioOutput("verifyingInboundPort", MESSAGES.verifyingInboundPort),
  new ScenarioAction(
    "verifyInboundPort",
    /**
     *
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup}} state
     */
    async (state) => {
      const client = new EC2Client({});
      const { SecurityGroups } = await client.send(
        new DescribeSecurityGroupsCommand({
          Filters: [{ Name: "group-name", Values: ["default"] }],
        }),
      );
      if (!SecurityGroups) {
        state.verifyInboundPortError = new Error(MESSAGES.noSecurityGroups);
      }
      state.defaultSecurityGroup = SecurityGroups[0];

      /**
       * @type {string}
       */
      const ipResponse = (await axios.get("http://checkip.amazonaws.com")).data;
      state.myIp = ipResponse.trim();
      const myIpRules = state.defaultSecurityGroup.IpPermissions.filter(
        ({ IpRanges }) =>
          IpRanges.some(
            ({ CidrIp }) =>
              CidrIp.startsWith(state.myIp) || CidrIp === "0.0.0.0/0",
          ),
      )
        .filter(({ IpProtocol }) => IpProtocol === "tcp")
        .filter(({ FromPort }) => FromPort === 80);

      state.myIpRules = myIpRules;
    },
  ),
  new ScenarioOutput(
    "verifiedInboundPort",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
```

```
      if (state.myIpRules.length > 0) {
        return MESSAGES.foundIpRules.replace(
          "${IP_RULES}",
          JSON.stringify(state.myIpRules, null, 2),
        );
      } else {
        return MESSAGES.noIpRules;
      }
    },
  ),
  new ScenarioInput(
    "shouldAddInboundRule",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return false;
      } else {
        return MESSAGES.noIpRules;
      }
    },
    { type: "confirm" },
  ),
  new ScenarioAction(
    "addInboundRule",
    /**
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup }} state
     */
    async (state) => {
      if (!state.shouldAddInboundRule) {
        return;
      }

      const client = new EC2Client({});
      await client.send(
        new AuthorizeSecurityGroupIngressCommand({
          GroupId: state.defaultSecurityGroup.GroupId,
          CidrIp: `${state.myIp}/32`,
          FromPort: 80,
          ToPort: 80,
          IpProtocol: "tcp",
        }),
```

```
      );
    },
  ),
  new ScenarioOutput("addedInboundRule", (state) => {
    if (state.shouldAddInboundRule) {
      return MESSAGES.addedInboundRule.replace("${IP_ADDRESS}", state.myIp);
    } else {
      return false;
    }
  }),
  new ScenarioOutput("verifyingEndpoint", (state) =>
    MESSAGES.verifyingEndpoint.replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioAction("verifyEndpoint", async (state) => {
    try {
      const response = await retry({ intervalInMs: 2000, maxRetries: 30 }, () =>
        axios.get(`http://${state.loadBalancerDns}`),
      );
      state.endpointResponse = JSON.stringify(response.data, null, 2);
    } catch (e) {
      state.verifyEndpointError = e;
    }
  }),
  new ScenarioOutput("verifiedEndpoint", (state) => {
    if (state.verifyEndpointError) {
      console.error(state.verifyEndpointError);
    } else {
      return MESSAGES.verifiedEndpoint.replace(
        "${ENDPOINT_RESPONSE}",
        state.endpointResponse,
      );
    }
  }),
];
```

Créez des étapes pour exécuter la démo.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { readFileSync } from "node:fs";
import { join } from "node:path";
```

```
import axios from "axios";

import {
  DescribeTargetGroupsCommand,
  DescribeTargetHealthCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";
import {
  DescribeInstanceInformationCommand,
  PutParameterCommand,
  SSMClient,
  SendCommandCommand,
} from "@aws-sdk/client-ssm";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DescribeAutoScalingGroupsCommand,
  TerminateInstanceInAutoScalingGroupCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  DescribeIamInstanceProfileAssociationsCommand,
  EC2Client,
  RebootInstancesCommand,
  ReplaceIamInstanceProfileAssociationCommand,
} from "@aws-sdk/client-ec2";

import {
  ScenarioAction,
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/scenario.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH } from "./constants.js";
import { findLoadBalancer } from "./shared.js";
```

```
const getRecommendation = new ScenarioAction(
  "getRecommendation",
  async (state) => {
    const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
    if (loadBalancer) {
      state.loadBalancerDnsName = loadBalancer.DNSName;
      try {
        state.recommendation = (
          await axios.get(`http://${state.loadBalancerDnsName}`)
        ).data;
      } catch (e) {
        state.recommendation = e instanceof Error ? e.message : e;
      }
    } else {
      throw new Error(MESSAGES.demoFindLoadBalancerError);
    }
  },
);

const getRecommendationResult = new ScenarioOutput(
  "getRecommendationResult",
  (state) =>
    `Recommendation:\n${JSON.stringify(state.recommendation, null, 2)}`,
  { preformatted: true },
);

const getHealthCheck = new ScenarioAction("getHealthCheck", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetGroups]
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetGroups]

  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  state.targetHealthDescriptions = TargetHealthDescriptions;
```

```
});

const getHealthCheckResult = new ScenarioOutput(
  "getHealthCheckResult",
  /**
   * @param {{ targetHealthDescriptions: import('@aws-sdk/client-elastic-load-
balancing-v2').TargetHealthDescription[]}} state
   */
  (state) => {
    const status = state.targetHealthDescriptions
      .map((th) => `${th.Target.Id}: ${th.TargetHealth.State}`)
      .join("\n");
    return `Health check:\n${status}`;
  },
  { preformatted: true },
);

const loadBalancerLoop = new ScenarioAction(
  "loadBalancerLoop",
  getRecommendation.action,
  {
    whileConfig: {
      whileFn: ({ loadBalancerCheck }) => loadBalancerCheck,
      input: new ScenarioInput(
        "loadBalancerCheck",
        MESSAGES.demoLoadBalancerCheck,
        {
          type: "confirm",
        },
      ),
      output: getRecommendationResult,
    },
  },
);

const healthCheckLoop = new ScenarioAction(
  "healthCheckLoop",
  getHealthCheck.action,
  {
    whileConfig: {
      whileFn: ({ healthCheck }) => healthCheck,
      input: new ScenarioInput("healthCheck", MESSAGES.demoHealthCheck, {
        type: "confirm",
      }),
```

```
      output: getHealthCheckResult,
    },
  },
);

const statusSteps = [
  getRecommendation,
  getRecommendationResult,
  getHealthCheck,
  getHealthCheckResult,
];

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const demoSteps = [
  new ScenarioOutput("header", MESSAGES.demoHeader, { header: true }),
  new ScenarioOutput("sanityCheck", MESSAGES.demoSanityCheck),
  ...statusSteps,
  new ScenarioInput(
    "brokenDependencyConfirmation",
    MESSAGES.demoBrokenDependencyConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("brokenDependency", async (state) => {
    if (!state.brokenDependencyConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      state.badTableName = `fake-table-${Date.now()}`;
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmTableNameKey,
          Value: state.badTableName,
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testBrokenDependency", (state) =>
    MESSAGES.demoTestBrokenDependency.replace(
      "${TABLE_NAME}",
      state.badTableName,
```

```
    ),
  ),
  ...statusSteps,
  new ScenarioInput(
    "staticResponseConfirmation",
    MESSAGES.demoStaticResponseConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("staticResponse", async (state) => {
    if (!state.staticResponseConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmFailureResponseKey,
          Value: "static",
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testStaticResponse", MESSAGES.demoTestStaticResponse),
  ...statusSteps,
  new ScenarioInput(
    "badCredentialsConfirmation",
    MESSAGES.demoBadCredentialsConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("badCredentialsExit", (state) => {
    if (!state.badCredentialsConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("fixDynamoDBName", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
```

```
    );
  }),
  new ScenarioAction(
    "badCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-auto-
scaling').Instance }} state
     */
    async (state) => {
      await createSsmOnlyInstanceProfile();
      const autoScalingClient = new AutoScalingClient({});
      const { AutoScalingGroups } = await autoScalingClient.send(
        new DescribeAutoScalingGroupsCommand({
          AutoScalingGroupNames: [NAMES.autoScalingGroupName],
        }),
      );
      state.targetInstance = AutoScalingGroups[0].Instances[0];
      // snippet-start:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
      const ec2Client = new EC2Client({});
      const { IamInstanceProfileAssociations } = await ec2Client.send(
        new DescribeIamInstanceProfileAssociationsCommand({
          Filters: [
            { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
          ],
        }),
      );
      // snippet-end:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
      state.instanceProfileAssociationId =
        IamInstanceProfileAssociations[0].AssociationId;
      // snippet-start:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]
      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        ec2Client.send(
          new ReplaceIamInstanceProfileAssociationCommand({
            AssociationId: state.instanceProfileAssociationId,
            IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
          }),
        ),
      );
      // snippet-end:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]
```

```
      await ec2Client.send(
        new RebootInstancesCommand({
          InstanceIds: [state.targetInstance.InstanceId],
        }),
      );

      const ssmClient = new SSMClient({});
      await retry({ intervalInMs: 20000, maxRetries: 15 }, async () => {
        const { InstanceInformationList } = await ssmClient.send(
          new DescribeInstanceInformationCommand({}),
        );

        const instance = InstanceInformationList.find(
          (info) => info.InstanceId === state.targetInstance.InstanceId,
        );

        if (!instance) {
          throw new Error("Instance not found.");
        }
      });

      await ssmClient.send(
        new SendCommandCommand({
          InstanceIds: [state.targetInstance.InstanceId],
          DocumentName: "AWS-RunShellScript",
          Parameters: { commands: ["cd / && sudo python3 server.py 80"] },
        }),
      );
    },
  ),
  new ScenarioOutput(
    "testBadCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation}} state
     */
    (state) =>
      MESSAGES.demoTestBadCredentials.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
  ),
  loadBalancerLoop,
  new ScenarioInput(
```

```
    "deepHealthCheckConfirmation",
    MESSAGES.demoDeepHealthCheckConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("deepHealthCheckExit", (state) => {
    if (!state.deepHealthCheckConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("deepHealthCheck", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmHealthCheckKey,
        Value: "deep",
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testDeepHealthCheck", MESSAGES.demoTestDeepHealthCheck),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "killInstanceConfirmation",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    (state) =>
      MESSAGES.demoKillInstanceConfirmation.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
    { type: "confirm" },
  ),
  new ScenarioAction("killInstanceExit", (state) => {
    if (!state.killInstanceConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction(
    "killInstance",
    /**
```

```
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    async (state) => {
      const client = new AutoScalingClient({});
      await client.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: state.targetInstance.InstanceId,
          ShouldDecrementDesiredCapacity: false,
        }),
      );
    },
  ),
  new ScenarioOutput("testKillInstance", MESSAGES.demoTestKillInstance),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput("failOpenConfirmation", MESSAGES.demoFailOpenConfirmation, {
    type: "confirm",
  }),
  new ScenarioAction("failOpenExit", (state) => {
    if (!state.failOpenConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("failOpen", () => {
    const client = new SSMClient({});
    return client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: `fake-table-${Date.now()}`,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testFailOpen", MESSAGES.demoFailOpenTest),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "resetTableConfirmation",
    MESSAGES.demoResetTableConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("resetTableExit", (state) => {
```

```
    if (!state.resetTableConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("resetTable", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testResetTable", MESSAGES.demoTestResetTable),
  healthCheckLoop,
  loadBalancerLoop,
];

async function createSsmOnlyInstanceProfile() {
  const iamClient = new IAMClient({});
  const { Policy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyName: NAMES.ssmOnlyPolicyName,
      PolicyDocument: readFileSync(
        join(RESOURCES_PATH, "ssm_only_policy.json"),
      ),
    }),
  );
  await iamClient.send(
    new CreateRoleCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      AssumeRolePolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
            Principal: { Service: "ec2.amazonaws.com" },
            Action: "sts:AssumeRole",
          },
        ],
      }),
    }),
```

```
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: Policy.Arn,
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
    }),
  );
  // snippet-start:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
  // snippet-end:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  await iamClient.send(
    new AddRoleToInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
      RoleName: NAMES.ssmOnlyRoleName,
    }),
  );

  return InstanceProfile;
}
```

Créez des étapes pour déployer toutes les ressources.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { unlinkSync } from "node:fs";

import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
```

```
  EC2Client,
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  DeleteInstanceProfileCommand,
  RemoveRoleFromInstanceProfileCommand,
  DeletePolicyCommand,
  DeleteRoleCommand,
  DetachRolePolicyCommand,
  paginateListPolicies,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DeleteAutoScalingGroupCommand,
  TerminateInstanceInAutoScalingGroupCommand,
  UpdateAutoScalingGroupCommand,
  paginateDescribeAutoScalingGroups,
} from "@aws-sdk/client-auto-scaling";
import {
  DeleteLoadBalancerCommand,
  DeleteTargetGroupCommand,
  DescribeTargetGroupsCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const destroySteps = [
  new ScenarioInput("destroy", MESSAGES.destroy, { type: "confirm" }),
  new ScenarioAction(
    "abort",
```

```
    (state) => state.destroy === false && process.exit(),
  ),
  new ScenarioAction("deleteTable", async (c) => {
    try {
      const client = new DynamoDBClient({});
      await client.send(new DeleteTableCommand({ TableName: NAMES.tableName }));
    } catch (e) {
      c.deleteTableError = e;
    }
  }),
  new ScenarioOutput("deleteTableResult", (state) => {
    if (state.deleteTableError) {
      console.error(state.deleteTableError);
      return MESSAGES.deleteTableError.replace(
        "${TABLE_NAME}",
        NAMES.tableName,
      );
    } else {
      return MESSAGES.deletedTable.replace("${TABLE_NAME}", NAMES.tableName);
    }
  }),
  new ScenarioAction("deleteKeyPair", async (state) => {
    try {
      const client = new EC2Client({});
      await client.send(
        new DeleteKeyPairCommand({ KeyName: NAMES.keyPairName }),
      );
      unlinkSync(`${NAMES.keyPairName}.pem`);
    } catch (e) {
      state.deleteKeyPairError = e;
    }
  }),
  new ScenarioOutput("deleteKeyPairResult", (state) => {
    if (state.deleteKeyPairError) {
      console.error(state.deleteKeyPairError);
      return MESSAGES.deleteKeyPairError.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
    } else {
      return MESSAGES.deletedKeyPair.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
```

```
    }
  }),
  new ScenarioAction("detachPolicyFromRole", async (state) => {
    try {
      const client = new IAMClient({});
      const policy = await findPolicy(NAMES.instancePolicyName);

      if (!policy) {
        state.detachPolicyFromRoleError = new Error(
          `Policy ${NAMES.instancePolicyName} not found.`,
        );
      } else {
        await client.send(
          new DetachRolePolicyCommand({
            RoleName: NAMES.instanceRoleName,
            PolicyArn: policy.Arn,
          }),
        );
      }
    } catch (e) {
      state.detachPolicyFromRoleError = e;
    }
  }),
  new ScenarioOutput("detachedPolicyFromRole", (state) => {
    if (state.detachPolicyFromRoleError) {
      console.error(state.detachPolicyFromRoleError);
      return MESSAGES.detachPolicyFromRoleError
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    } else {
      return MESSAGES.detachedPolicyFromRole
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
  }),
  new ScenarioAction("deleteInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const policy = await findPolicy(NAMES.instancePolicyName);

    if (!policy) {
      state.deletePolicyError = new Error(
        `Policy ${NAMES.instancePolicyName} not found.`,
      );
    } else {
```

```
      return client.send(
        new DeletePolicyCommand({
          PolicyArn: policy.Arn,
        }),
      );
    }
  }),
  new ScenarioOutput("deletePolicyResult", (state) => {
    if (state.deletePolicyError) {
      console.error(state.deletePolicyError);
      return MESSAGES.deletePolicyError.replace(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    } else {
      return MESSAGES.deletedPolicy.replace(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    }
  }),
  new ScenarioAction("removeRoleFromInstanceProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          RoleName: NAMES.instanceRoleName,
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.removeRoleFromInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("removeRoleFromInstanceProfileResult", (state) => {
    if (state.removeRoleFromInstanceProfile) {
      console.error(state.removeRoleFromInstanceProfileError);
      return MESSAGES.removeRoleFromInstanceProfileError
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    } else {
      return MESSAGES.removedRoleFromInstanceProfile
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
```

```
    }
  }),
  new ScenarioAction("deleteInstanceRole", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteRoleCommand({
          RoleName: NAMES.instanceRoleName,
        }),
      );
    } catch (e) {
      state.deleteInstanceRoleError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceRoleResult", (state) => {
    if (state.deleteInstanceRoleError) {
      console.error(state.deleteInstanceRoleError);
      return MESSAGES.deleteInstanceRoleError.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    } else {
      return MESSAGES.deletedInstanceRole.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    }
  }),
  new ScenarioAction("deleteInstanceProfile", async (state) => {
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
      const client = new IAMClient({});
      await client.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
    } catch (e) {
      state.deleteInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceProfileResult", (state) => {
    if (state.deleteInstanceProfileError) {
```

```
      console.error(state.deleteInstanceProfileError);
      return MESSAGES.deleteInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    } else {
      return MESSAGES.deletedInstanceProfile.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    }
  }),
  new ScenarioAction("deleteAutoScalingGroup", async (state) => {
    try {
      await terminateGroupInstances(NAMES.autoScalingGroupName);
      await retry({ intervalInMs: 60000, maxRetries: 60 }, async () => {
        await deleteAutoScalingGroup(NAMES.autoScalingGroupName);
      });
    } catch (e) {
      state.deleteAutoScalingGroupError = e;
    }
  }),
  new ScenarioOutput("deleteAutoScalingGroupResult", (state) => {
    if (state.deleteAutoScalingGroupError) {
      console.error(state.deleteAutoScalingGroupError);
      return MESSAGES.deleteAutoScalingGroupError.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    } else {
      return MESSAGES.deletedAutoScalingGroup.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    }
  }),
  new ScenarioAction("deleteLaunchTemplate", async (state) => {
    const client = new EC2Client({});
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
      await client.send(
        new DeleteLaunchTemplateCommand({
          LaunchTemplateName: NAMES.launchTemplateName,
        }),
```

```
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
    } catch (e) {
      state.deleteLaunchTemplateError = e;
    }
  }),
  new ScenarioOutput("deleteLaunchTemplateResult", (state) => {
    if (state.deleteLaunchTemplateError) {
      console.error(state.deleteLaunchTemplateError);
      return MESSAGES.deleteLaunchTemplateError.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    } else {
      return MESSAGES.deletedLaunchTemplate.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    }
  }),
  new ScenarioAction("deleteLoadBalancer", async (state) => {
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
      const client = new ElasticLoadBalancingV2Client({});
      const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
      await client.send(
        new DeleteLoadBalancerCommand({
          LoadBalancerArn: loadBalancer.LoadBalancerArn,
        }),
      );
      await retry({ intervalInMs: 1000, maxRetries: 60 }, async () => {
        const lb = await findLoadBalancer(NAMES.loadBalancerName);
        if (lb) {
          throw new Error("Load balancer still exists.");
        }
      });
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
    } catch (e) {
      state.deleteLoadBalancerError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerResult", (state) => {
    if (state.deleteLoadBalancerError) {
      console.error(state.deleteLoadBalancerError);
```

```
      return MESSAGES.deleteLoadBalancerError.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    } else {
      return MESSAGES.deletedLoadBalancer.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    }
  }),
  new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
    const client = new ElasticLoadBalancingV2Client({});
    try {
      const { TargetGroups } = await client.send(
        new DescribeTargetGroupsCommand({
          Names: [NAMES.loadBalancerTargetGroupName],
        }),
      );

      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        client.send(
          new DeleteTargetGroupCommand({
            TargetGroupArn: TargetGroups[0].TargetGroupArn,
          }),
        ),
      );
    } catch (e) {
      state.deleteLoadBalancerTargetGroupError = e;
    }
    // snippet-end:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
  }),
  new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
    if (state.deleteLoadBalancerTargetGroupError) {
      console.error(state.deleteLoadBalancerTargetGroupError);
      return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
      );
    } else {
      return MESSAGES.deletedLoadBalancerTargetGroup.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
```

```
      );
    }
  }),
  new ScenarioAction("detachSsmOnlyRoleFromProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyRoleFromProfileError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyRoleFromProfileResult", (state) => {
    if (state.detachSsmOnlyRoleFromProfileError) {
      console.error(state.detachSsmOnlyRoleFromProfileError);
      return MESSAGES.detachSsmOnlyRoleFromProfileError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    } else {
      return MESSAGES.detachedSsmOnlyRoleFromProfile
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    }
  }),
  new ScenarioAction("detachSsmOnlyCustomRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyCustomRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyCustomRolePolicyResult", (state) => {
    if (state.detachSsmOnlyCustomRolePolicyError) {
```

```
      console.error(state.detachSsmOnlyCustomRolePolicyError);
      return MESSAGES.detachSsmOnlyCustomRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    } else {
      return MESSAGES.detachedSsmOnlyCustomRolePolicy
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    }
  }),
  new ScenarioAction("detachSsmOnlyAWSRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
        }),
      );
    } catch (e) {
      state.detachSsmOnlyAWSRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyAWSRolePolicyResult", (state) => {
    if (state.detachSsmOnlyAWSRolePolicyError) {
      console.error(state.detachSsmOnlyAWSRolePolicyError);
      return MESSAGES.detachSsmOnlyAWSRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    } else {
      return MESSAGES.detachedSsmOnlyAWSRolePolicy
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    }
  }),
  new ScenarioAction("deleteSsmOnlyInstanceProfile", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
        }),
      );
    } catch (e) {
```

```
      state.deleteSsmOnlyInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyInstanceProfileResult", (state) => {
    if (state.deleteSsmOnlyInstanceProfileError) {
      console.error(state.deleteSsmOnlyInstanceProfileError);
      return MESSAGES.deleteSsmOnlyInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    }
  }),
  new ScenarioAction("deleteSsmOnlyPolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DeletePolicyCommand({
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyPolicyError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyPolicyResult", (state) => {
    if (state.deleteSsmOnlyPolicyError) {
      console.error(state.deleteSsmOnlyPolicyError);
      return MESSAGES.deleteSsmOnlyPolicyError.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyPolicy.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    }
  }),
```

```
  new ScenarioAction("deleteSsmOnlyRole", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteRoleCommand({
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyRoleError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyRoleResult", (state) => {
    if (state.deleteSsmOnlyRoleError) {
      console.error(state.deleteSsmOnlyRoleError);
      return MESSAGES.deleteSsmOnlyRoleError.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyRole.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    }
  }),
];

/**
 * @param {string} policyName
 */
async function findPolicy(policyName) {
  const client = new IAMClient({});
  const paginatedPolicies = paginateListPolicies({ client }, {});
  for await (const page of paginatedPolicies) {
    const policy = page.Policies.find((p) => p.PolicyName === policyName);
    if (policy) {
      return policy;
    }
  }
}

/**
 * @param {string} groupName
```

```
 */
async function deleteAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  try {
    await client.send(
      new DeleteAutoScalingGroupCommand({
        AutoScalingGroupName: groupName,
      }),
    );
  } catch (err) {
    if (!(err instanceof Error)) {
      throw err;
    } else {
      console.log(err.name);
      throw err;
    }
  }
}

/**
 * @param {string} groupName
 */
async function terminateGroupInstances(groupName) {
  const autoScalingClient = new AutoScalingClient({});
  const group = await findAutoScalingGroup(groupName);
  await autoScalingClient.send(
    new UpdateAutoScalingGroupCommand({
      AutoScalingGroupName: group.AutoScalingGroupName,
      MinSize: 0,
    }),
  );
  for (const i of group.Instances) {
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: i.InstanceId,
          ShouldDecrementDesiredCapacity: true,
        }),
      ),
    );
  }
}

async function findAutoScalingGroup(groupName) {
```

```
  const client = new AutoScalingClient({});
  const paginatedGroups = paginateDescribeAutoScalingGroups({ client }, {});
  for await (const page of paginatedGroups) {
    const group = page.AutoScalingGroups.find(
      (g) => g.AutoScalingGroupName === groupName,
    );
    if (group) {
      return group;
    }
  }
  throw new Error(`Auto scaling group ${groupName} not found.`);
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK for JavaScript .
  - AttachLoadBalancerTargetGroupes
  - CreateAutoScalingGroup
  - CreateInstanceProfil
  - CreateLaunchModèle
  - CreateListener
  - CreateLoadÉquilibreur
  - CreateTargetGroupe
  - DeleteAutoScalingGroup
  - DeleteInstanceProfil
  - DeleteLaunchModèle
  - DeleteLoadÉquilibreur
  - DeleteTargetGroupe
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadÉquilibreurs
  - DescribeSubnets
  - DescribeTargetGroupes

- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Exécutez un scénario interactif à une invite de commande.

```
class Runner:
    def __init__(
        self, resource_path, recommendation, autoscaler, loadbalancer,
 param_helper
    ):
        self.resource_path = resource_path
        self.recommendation = recommendation
        self.autoscaler = autoscaler
        self.loadbalancer = loadbalancer
        self.param_helper = param_helper
        self.protocol = "HTTP"
        self.port = 80
        self.ssh_port = 22

    def deploy(self):
        recommendations_path = f"{self.resource_path}/recommendations.json"
        startup_script = f"{self.resource_path}/server_startup_script.sh"
        instance_policy = f"{self.resource_path}/instance_policy.json"

        print(
```

```
            "\nFor this demo, we'll use the AWS SDK for Python (Boto3) to create
several AWS resources\n"
            "to set up a load-balanced web service endpoint and explore some ways
to make it resilient\n"
            "against various kinds of failures.\n\n"
            "Some of the resources create by this demo are:\n"
        )
        print(
            "\t* A DynamoDB table that the web service depends on to provide
book, movie, and song recommendations."
        )
        print(
            "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server."
        )
        print(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones."
        )
        print(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests."
        )
        print("-" * 88)
        q.ask("Press Enter when you're ready to start deploying resources.")

        print(
            f"Creating and populating a DynamoDB table named
'{self.recommendation.table_name}'."
        )
        self.recommendation.create()
        self.recommendation.populate(recommendations_path)
        print("-" * 88)

        print(
            f"Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.\n"
            f"This script starts a Python web server defined in the `server.py`
script. The web server\n"
            f"listens to HTTP requests on port 80 and responds to requests to '/'
and to '/healthcheck'.\n"
            f"For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
```

```
            f"run a web server, such as Apache, with least-privileged
credentials.\n"
        )
        print(
            f"The template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            f"permissions to access the DynamoDB recommendation table and Systems
Manager parameters\n"
            f"that control the flow of the demo.\n"
        )
        self.autoscaler.create_template(startup_script, instance_policy)
        print("-" * 88)

        print(
            f"Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            f"Availability Zone."
        )
        zones = self.autoscaler.create_group(3)
        print("-" * 88)
        print(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
            "HTTP requests. You can see these instances in the console or
continue with the demo."
        )
        print("-" * 88)
        q.ask("Press Enter when you're ready to continue.")

        print(f"Creating variables that control the flow of the demo.\n")
        self.param_helper.reset()

        print(
            "\nCreating an Elastic Load Balancing target group and load balancer.
The target group\n"
            "defines how the load balancer connects to instances. The load
balancer provides a\n"
            "single endpoint where clients connect and dispatches requests to
instances in the group.\n"
        )
        vpc = self.autoscaler.get_default_vpc()
        subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
        target_group = self.loadbalancer.create_target_group(
            self.protocol, self.port, vpc["VpcId"]
```

```
        )
        self.loadbalancer.create_load_balancer(
            [subnet["SubnetId"] for subnet in subnets], target_group
        )
        self.autoscaler.attach_load_balancer_target_group(target_group)
        print(f"Verifying access to the load balancer endpoint...")
        lb_success = self.loadbalancer.verify_load_balancer_endpoint()
        if not lb_success:
            print(
                "Couldn't connect to the load balancer, verifying that the port
 is open..."
            )
            current_ip_address = requests.get(
                "http://checkip.amazonaws.com"
            ).text.strip()
            sec_group, port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.port, current_ip_address
            )
            sec_group, ssh_port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.ssh_port, current_ip_address
            )
            if not port_is_open:
                print(
                    "For this example to work, the default security group for
 your default VPC must\n"
                    "allows access from this computer. You can either add it
 automatically from this\n"
                    "example or add it yourself using the AWS Management Console.
\n"
                )
                if q.ask(
                    f"Do you want to add a rule to security group
 {sec_group['GroupId']} to allow\n"
                    f"inbound traffic on port {self.port} from your computer's IP
 address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.port, current_ip_address
                    )
            if not ssh_port_is_open:
                if q.ask(
                    f"Do you want to add a rule to security group
 {sec_group['GroupId']} to allow\n"
```

```
                    f"inbound SSH traffic on port {self.ssh_port} for debugging
 from your computer's IP address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.ssh_port, current_ip_address
                    )
            lb_success = self.loadbalancer.verify_load_balancer_endpoint()
        if lb_success:
            print("Your load balancer is ready. You can access it by browsing to:
\n")
            print(f"\thttp://{self.loadbalancer.endpoint()}\n")
        else:
            print(
                "Couldn't get a successful response from the load balancer
 endpoint. Troubleshoot by\n"
                "manually verifying that your VPC and security group are
 configured correctly and that\n"
                "you can successfully make a GET request to the load balancer
 endpoint:\n"
            )
            print(f"\thttp://{self.loadbalancer.endpoint()}\n")
        print("-" * 88)
        q.ask("Press Enter when you're ready to continue with the demo.")

    def demo_choices(self):
        actions = [
            "Send a GET request to the load balancer endpoint.",
            "Check the health of load balancer targets.",
            "Go to the next part of the demo.",
        ]
        choice = 0
        while choice != 2:
            print("-" * 88)
            print(
                "\nSee the current state of the service by selecting one of the
 following choices:\n"
            )
            choice = q.choose("\nWhich action would you like to take? ", actions)
            print("-" * 88)
            if choice == 0:
                print("Request:\n")
                print(f"GET http://{self.loadbalancer.endpoint()}")
                response = requests.get(f"http://{self.loadbalancer.endpoint()}")
```

```
                print("\nResponse:\n")
                print(f"{response.status_code}")
                if response.headers.get("content-type") == "application/json":
                    pp(response.json())
            elif choice == 1:
                print("\nChecking the health of load balancer targets:\n")
                health = self.loadbalancer.check_target_health()
                for target in health:
                    state = target["TargetHealth"]["State"]
                    print(
                        f"\tTarget {target['Target']['Id']} on port
{target['Target']['Port']} is {state}"
                    )
                    if state != "healthy":
                        print(
                            f"\t\t{target['TargetHealth']['Reason']}:
{target['TargetHealth']['Description']}\n"
                        )
                print(
                    f"\nNote that it can take a minute or two for the health
check to update\n"
                    f"after changes are made.\n"
                )
            elif choice == 2:
                print("\nOkay, let's move on.")
                print("-" * 88)

    def demo(self):
        ssm_only_policy = f"{self.resource_path}/ssm_only_policy.json"

        print("\nResetting parameters to starting values for demo.\n")
        self.param_helper.reset()

        print(
            "\nThis part of the demonstration shows how to toggle different parts
of the system\n"
            "to create situations where the web service fails, and shows how
using a resilient\n"
            "architecture can keep the web service running in spite of these
failures."
        )
        print("-" * 88)

        print(
```

```
            "At the start, the load balancer endpoint returns recommendations and
reports that all targets are healthy."
        )
        self.demo_choices()

        print(
            f"The web service running on the EC2 instances gets recommendations
by querying a DynamoDB table.\n"
            f"The table name is contained in a Systems Manager parameter named
'{self.param_helper.table}'.\n"
            f"To simulate a failure of the recommendation service, let's set this
parameter to name a non-existent table.\n"
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        print(
            "\nNow, sending a GET request to the load balancer endpoint returns a
failure code. But, the service reports as\n"
            "healthy to the load balancer because shallow health checks don't
check for failure of the recommendation service."
        )
        self.demo_choices()

        print(
            f"Instead of failing when the recommendation service fails, the web
service can return a static response.\n"
            f"While this is not a perfect solution, it presents the customer with
a somewhat better experience than failure.\n"
        )
        self.param_helper.put(self.param_helper.failure_response, "static")
        print(
            f"\nNow, sending a GET request to the load balancer endpoint returns
a static response.\n"
            f"The service still reports as healthy because health checks are
still shallow.\n"
        )
        self.demo_choices()

        print("Let's reinstate the recommendation service.\n")
        self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)
        print(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n"
            "access the DynamoDB recommendation table.\n"
```

```
        )
        self.autoscaler.create_instance_profile(
            ssm_only_policy,
            self.autoscaler.bad_creds_policy_name,
            self.autoscaler.bad_creds_role_name,
            self.autoscaler.bad_creds_profile_name,
            ["AmazonSSMManagedInstanceCore"],
        )
        instances = self.autoscaler.get_instances()
        bad_instance_id = instances[0]
        instance_profile = self.autoscaler.get_instance_profile(bad_instance_id)
        print(
            f"\nReplacing the profile for instance {bad_instance_id} with a
profile that contains\n"
            f"bad credentials...\n"
        )
        self.autoscaler.replace_instance_profile(
            bad_instance_id,
            self.autoscaler.bad_creds_profile_name,
            instance_profile["AssociationId"],
        )
        print(
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n"
            "depending on which instance is selected by the load balancer.\n"
        )
        self.demo_choices()

        print(
            "\nLet's implement a deep health check. For this demo, a deep health
check tests whether\n"
            "the web service can access the DynamoDB table that it depends on for
recommendations. Note that\n"
            "the deep health check is only for ELB routing and not for Auto
Scaling instance health.\n"
            "This kind of deep health check is not recommended for Auto Scaling
instance health, because it\n"
            "risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.\n"
        )
        print(
            "By implementing deep health checks, the load balancer can detect
when one of the instances is failing\n"
            "and take that instance out of rotation.\n"
```

```
        )
        self.param_helper.put(self.param_helper.health_check, "deep")
        print(
            f"\nNow, checking target health indicates that the instance with bad
credentials ({bad_instance_id})\n"
            f"is unhealthy. Note that it might take a minute or two for the load
balancer to detect the unhealthy \n"
            f"instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because\n"
            "the load balancer takes unhealthy instances out of its rotation.\n"
        )
        self.demo_choices()

        print(
            "\nBecause the instances in this demo are controlled by an auto
scaler, the simplest way to fix an unhealthy\n"
            "instance is to terminate it and let the auto scaler start a new
instance to replace it.\n"
        )
        self.autoscaler.terminate_instance(bad_instance_id)
        print(
            "\nEven while the instance is terminating and the new instance is
starting, sending a GET\n"
            "request to the web service continues to get a successful
recommendation response because\n"
            "the load balancer routes requests to the healthy instances. After
the replacement instance\n"
            "starts and reports as healthy, it is included in the load balancing
rotation.\n"
            "\nNote that terminating and replacing an instance typically takes
several minutes, during which time you\n"
            "can see the changing health check status until the new instance is
running and healthy.\n"
        )
        self.demo_choices()

        print(
            "\nIf the recommendation service fails now, deep health checks mean
all instances report as unhealthy.\n"
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        print(
            "\nWhen all instances are unhealthy, the load balancer continues to
route requests even to\n"
```

```
            "unhealthy instances, allowing them to fail open and return a static
 response rather than fail\n"
            "closed and report failure to the customer."
        )
        self.demo_choices()
        self.param_helper.reset()

    def destroy(self):
        print(
            "This concludes the demo of how to build and manage a resilient
 service.\n"
            "To keep things tidy and to avoid unwanted charges on your account,
 we can clean up all AWS resources\n"
            "that were created for this demo."
        )
        if q.ask("Do you want to clean up all demo resources? (y/n) ",
 q.is_yesno):
            self.loadbalancer.delete_load_balancer()
            self.loadbalancer.delete_target_group()
            self.autoscaler.delete_group()
            self.autoscaler.delete_key_pair()
            self.autoscaler.delete_template()
            self.autoscaler.delete_instance_profile(
                self.autoscaler.bad_creds_profile_name,
                self.autoscaler.bad_creds_role_name,
            )
            self.recommendation.destroy()
        else:
            print(
                "Okay, we'll leave the resources intact.\n"
                "Don't forget to delete them when you're done with them or you
 might incur unexpected charges."
            )


def main():
    parser = argparse.ArgumentParser()
    parser.add_argument(
        "--action",
        required=True,
        choices=["all", "deploy", "demo", "destroy"],
        help="The action to take for the demo. When 'all' is specified, resources
 are\n"
        "deployed, the demo is run, and resources are destroyed.",
```

```
    )
    parser.add_argument(
        "--resource_path",
        default="../../../workflows/resilient_service/resources",
        help="The path to resource files used by this example, such as IAM
 policies and\n"
        "instance scripts.",
    )
    args = parser.parse_args()

    print("-" * 88)
    print(
        "Welcome to the demonstration of How to Build and Manage a Resilient
 Service!"
    )
    print("-" * 88)

    prefix = "doc-example-resilience"
    recommendation = RecommendationService.from_client(
        "doc-example-recommendation-service"
    )
    autoscaler = AutoScaler.from_client(prefix)
    loadbalancer = LoadBalancer.from_client(prefix)
    param_helper = ParameterHelper.from_client(recommendation.table_name)
    runner = Runner(
        args.resource_path, recommendation, autoscaler, loadbalancer,
 param_helper
    )
    actions = [args.action] if args.action != "all" else ["deploy", "demo",
 "destroy"]
    for action in actions:
        if action == "deploy":
            runner.deploy()
        elif action == "demo":
            runner.demo()
        elif action == "destroy":
            runner.destroy()

    print("-" * 88)
    print("Thanks for watching!")
    print("-" * 88)


if __name__ == "__main__":
```

```
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    main()
```

Créez une classe qui englobe les actions Auto Scaling et Amazon EC2.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
```

```
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from Boto3 clients.

        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        """
        as_client = boto3.client("autoscaling")
        ec2_client = boto3.client("ec2")
        ssm_client = boto3.client("ssm")
        iam_client = boto3.client("iam")
        return cls(
            resource_prefix,
            "t3.micro",
            "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
            as_client,
            ec2_client,
            ssm_client,
            iam_client,
        )

    def create_instance_profile(
        self, policy_file, policy_name, role_name, profile_name,
aws_managed_policies=()
    ):
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
```

```
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        with open(policy_file) as file:
            instance_policy_doc = file.read()

        policy_arn = None
        try:
            pol_response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=instance_policy_doc
            )
            policy_arn = pol_response["Policy"]["Arn"]
            log.info("Created policy with ARN %s.", policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Policy %s already exists, nothing to do.", policy_name)
                list_pol_response = self.iam_client.list_policies(Scope="Local")
                for pol in list_pol_response["Policies"]:
                    if pol["PolicyName"] == policy_name:
                        policy_arn = pol["Arn"]
                        break
            if policy_arn is None:
                raise AutoScalerError(f"Couldn't create policy {policy_name}:
{err}")
```

```
        try:
            self.iam_client.create_role(
                RoleName=role_name,
AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            self.iam_client.attach_role_policy(RoleName=role_name,
PolicyArn=policy_arn)
            for aws_policy in aws_managed_policies:
                self.iam_client.attach_role_policy(
                    RoleName=role_name,
                    PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
                )
            log.info("Created role %s and attached policy %s.", role_name,
policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Role %s already exists, nothing to do.", role_name)
            else:
                raise AutoScalerError(f"Couldn't create role {role_name}: {err}")

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            else:
```

```
                raise AutoScalerError(
                    f"Couldn't create profile {profile_name} and attach it to
role\n"
                    f"{role_name}: {err}"
                )
        return profile_arn


    def get_instance_profile(self, instance_id):
        """
        Gets data about the profile associated with an instance.

        :param instance_id: The ID of the instance to look up.
        :return: The profile data.
        """
        try:
            response =
self.ec2_client.describe_iam_instance_profile_associations(
                Filters=[{"Name": "instance-id", "Values": [instance_id]}]
            )
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't get instance profile association for instance
{instance_id}: {err}"
            )
        else:
            return response["IamInstanceProfileAssociations"][0]


    def replace_instance_profile(
        self, instance_id, new_instance_profile_name, profile_association_id
    ):
        """
        Replaces the profile associated with a running instance. After the
profile is
        replaced, the instance is rebooted to ensure that it uses the new
profile. When
        the instance is ready, Systems Manager is used to restart the Python web
server.

        :param instance_id: The ID of the instance to update.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
```

```
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)
            inst_ready = False
            tries = 0
            while not inst_ready:
                if tries % 6 == 0:
                    self.ec2_client.reboot_instances(InstanceIds=[instance_id])
                    log.info(
                        "Rebooting instance %s and waiting for it to to be
ready.",
                        instance_id,
                    )
                tries += 1
                time.sleep(10)
                response = self.ssm_client.describe_instance_information()
                for info in response["InstanceInformationList"]:
                    if info["InstanceId"] == instance_id:
                        inst_ready = True
            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info("Restarted the Python web server on instance %s.",
instance_id)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't replace instance profile for association
{profile_association_id}: {err}"
            )
```

```
   def delete_instance_profile(self, profile_name, role_name):
       """
       Detaches a role from an instance profile, detaches policies from the
role,
       and deletes all the resources.

       :param profile_name: The name of the profile to delete.
       :param role_name: The name of the role to delete.
       """
       try:
           self.iam_client.remove_role_from_instance_profile(
               InstanceProfileName=profile_name, RoleName=role_name
           )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
           log.info("Deleted instance profile %s.", profile_name)
           attached_policies = self.iam_client.list_attached_role_policies(
               RoleName=role_name
           )
           for pol in attached_policies["AttachedPolicies"]:
               self.iam_client.detach_role_policy(
                   RoleName=role_name, PolicyArn=pol["PolicyArn"]
               )
               if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                   self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
               log.info("Detached and deleted policy %s.", pol["PolicyName"])
           self.iam_client.delete_role(RoleName=role_name)
           log.info("Deleted role %s.", role_name)
       except ClientError as err:
           if err.response["Error"]["Code"] == "NoSuchEntity":
               log.info(
                   "Instance profile %s doesn't exist, nothing to do.",
profile_name
               )
           else:
               raise AutoScalerError(
                   f"Couldn't delete instance profile {profile_name} or detach "
                   f"policies and delete role {role_name}: {err}"
               )


   def create_key_pair(self, key_pair_name):
       """
```

```
        Creates a new key pair.

        :param key_pair_name: The name of the key pair to create.
        :return: The newly created key pair.
        """
        try:
            response = self.ec2_client.create_key_pair(KeyName=key_pair_name)
            with open(f"{key_pair_name}.pem", "w") as file:
                file.write(response["KeyMaterial"])
            chmod(f"{key_pair_name}.pem", 0o600)
            log.info("Created key pair %s.", key_pair_name)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't create key pair {key_pair_name}:
{err}")


    def delete_key_pair(self):
        """
        Deletes a key pair.

        :param key_pair_name: The name of the key pair to delete.
        """
        try:
            self.ec2_client.delete_key_pair(KeyName=self.key_pair_name)
            remove(f"{self.key_pair_name}.pem")
            log.info("Deleted key pair %s.", self.key_pair_name)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't delete key pair {self.key_pair_name}: {err}"
            )
        except FileNotFoundError:
            log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
        except PermissionError:
            log.info(
                "Inadequate permissions to delete key pair %s.",
self.key_pair_name
            )
        except Exception as err:
            raise AutoScalerError(
                f"Couldn't delete key pair {self.key_pair_name}: {err}"
            )
```

```
    def create_template(self, server_startup_script_file, instance_policy_file):
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling. The
        launch template specifies a Bash script in its user data field that runs
after
        the instance is started. This script installs Python packages and starts
a
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
run
                                           when an instance starts.
        :param instance_policy_file: The path to a file that defines a
permissions policy
                                     to create and attach to the instance
profile.
        :return: Information about the newly created template.
        """
        template = {}
        try:
            self.create_key_pair(self.key_pair_name)
            self.create_instance_profile(
                instance_policy_file,
                self.instance_policy_name,
                self.instance_role_name,
                self.instance_profile_name,
            )
            with open(server_startup_script_file) as file:
                start_server_script = file.read()
            ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
            ami_id = ami_latest["Parameter"]["Value"]
            lt_response = self.ec2_client.create_launch_template(
                LaunchTemplateName=self.launch_template_name,
                LaunchTemplateData={
                    "InstanceType": self.inst_type,
                    "ImageId": ami_id,
                    "IamInstanceProfile": {"Name": self.instance_profile_name},
                    "UserData": base64.b64encode(
                        start_server_script.encode(encoding="utf-8")
                    ).decode(encoding="utf-8"),
                    "KeyName": self.key_pair_name,
                },
            )
```

```
            template = lt_response["LaunchTemplate"]
            log.info(
                "Created launch template %s for AMI %s on %s.",
                self.launch_template_name,
                ami_id,
                self.inst_type,
            )
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.AlreadyExistsException"
            ):
                log.info(
                    "Launch template %s already exists, nothing to do.",
                    self.launch_template_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create launch template
{self.launch_template_name}: {err}."
                )
        return template


    def delete_template(self):
        """
        Deletes a launch template.
        """
        try:
            self.ec2_client.delete_launch_template(
                LaunchTemplateName=self.launch_template_name
            )
            self.delete_instance_profile(
                self.instance_profile_name, self.instance_role_name
            )
            log.info("Launch template %s deleted.", self.launch_template_name)
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.NotFoundException"
            ):
                log.info(
                    "Launch template %s does not exist, nothing to do.",
                    self.launch_template_name,
```

```
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete launch template
{self.launch_template_name}: {err}."
                )


    def get_availability_zones(self):
        """
        Gets a list of Availability Zones in the AWS Region of the Amazon EC2
client.

        :return: The list of Availability Zones for the client Region.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get availability zones: {err}.")
        else:
            return zones


    def create_group(self, group_size):
        """
        Creates an EC2 Auto Scaling group with the specified size.

        :param group_size: The number of instances to set for the minimum and
maximum in
                           the group.
        :return: The list of Availability Zones specified for the group.
        """
        zones = []
        try:
            zones = self.get_availability_zones()
            self.autoscaling_client.create_auto_scaling_group(
                AutoScalingGroupName=self.group_name,
                AvailabilityZones=zones,
                LaunchTemplate={
                    "LaunchTemplateName": self.launch_template_name,
                    "Version": "$Default",
                },
                MinSize=group_size,
```

```
                MaxSize=group_size,
            )
            log.info(
                "Created EC2 Auto Scaling group %s with availability zones %s.",
                self.launch_template_name,
                zones,
            )
        except ClientError as err:
            if err.response["Error"]["Code"] == "AlreadyExists":
                log.info(
                    "EC2 Auto Scaling group %s already exists, nothing to do.",
                    self.group_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create EC2 Auto Scaling group {self.group_name}:
{err}"
                )
        return zones


    def get_instances(self):
        """
        Gets data about the instances in the EC2 Auto Scaling group.

        :return: Data about the instances.
        """
        try:
            as_response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            instance_ids = [
                i["InstanceId"]
                for i in as_response["AutoScalingGroups"][0]["Instances"]
            ]
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't get instances for Auto Scaling group
{self.group_name}: {err}"
            )
        else:
            return instance_ids
```

```
    def terminate_instance(self, instance_id):
        """
        Terminates and instances in an EC2 Auto Scaling group. After an instance
is
        terminated, it can no longer be accessed.

        :param instance_id: The ID of the instance to terminate.
        """
        try:
            self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=instance_id, ShouldDecrementDesiredCapacity=False
            )
            log.info("Terminated instance %s.", instance_id)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't terminate instance {instance_id}:
{err}")

    def attach_load_balancer_target_group(self, lb_target_group):
        """
        Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
Scaling group.
        The target group specifies how the load balancer forward requests to the
instances
        in the group.

        :param lb_target_group: Data about the ELB target group to attach.
        """
        try:
            self.autoscaling_client.attach_load_balancer_target_groups(
                AutoScalingGroupName=self.group_name,
                TargetGroupARNs=[lb_target_group["TargetGroupArn"]],
            )
            log.info(
                "Attached load balancer target group %s to auto scaling group
%s.",
                lb_target_group["TargetGroupName"],
                self.group_name,
            )
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't attach load balancer target group
{lb_target_group['TargetGroupName']}\n"
                f"to auto scaling group {self.group_name}"
            )
```

```
   def _try_terminate_instance(self, inst_id):
       stopping = False
       log.info(f"Stopping {inst_id}.")
       while not stopping:
           try:
               self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                   InstanceId=inst_id, ShouldDecrementDesiredCapacity=True
               )
               stopping = True
           except ClientError as err:
               if err.response["Error"]["Code"] == "ScalingActivityInProgress":
                   log.info("Scaling activity in progress for %s. Waiting...",
inst_id)
                   time.sleep(10)
               else:
                   raise AutoScalerError(f"Couldn't stop instance {inst_id}:
{err}.")

   def _try_delete_group(self):
       """
       Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
       the function waits and retries until the group is successfully deleted.
       """
       stopped = False
       while not stopped:
           try:
               self.autoscaling_client.delete_auto_scaling_group(
                   AutoScalingGroupName=self.group_name
               )
               stopped = True
               log.info("Deleted EC2 Auto Scaling group %s.", self.group_name)
           except ClientError as err:
               if (
                   err.response["Error"]["Code"] == "ResourceInUse"
                   or err.response["Error"]["Code"] ==
"ScalingActivityInProgress"
               ):
                   log.info(
                       "Some instances are still running. Waiting for them to
stop..."
                   )
```

```
                    time.sleep(10)
                else:
                    raise AutoScalerError(
                        f"Couldn't delete group {self.group_name}: {err}."
                    )

    def delete_group(self):
        """
        Terminates all instances in the group, deletes the EC2 Auto Scaling
 group.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            groups = response.get("AutoScalingGroups", [])
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
                    AutoScalingGroupName=self.group_name, MinSize=0
                )
                instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
                for inst_id in instance_ids:
                    self._try_terminate_instance(inst_id)
                self._try_delete_group()
            else:
                log.info("No groups found named %s, nothing to do.",
 self.group_name)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't delete group {self.group_name}:
 {err}.")


    def get_default_vpc(self):
        """
        Gets the default VPC for the account.

        :return: Data about the default VPC.
        """
        try:
            response = self.ec2_client.describe_vpcs(
                Filters=[{"Name": "is-default", "Values": ["true"]}]
            )
        except ClientError as err:
```

```
            raise AutoScalerError(f"Couldn't get default VPC: {err}")
        else:
            return response["Vpcs"][0]


    def verify_inbound_port(self, vpc, port, ip_address):
        """
        Verify the default security group of the specified VPC allows ingress
from this
        computer. This can be done by allowing ingress from this computer's IP
        address. In some situations, such as connecting from a corporate network,
you
        must instead specify a prefix list ID. You can also temporarily open the
port to
        any IP address while running this example. If you do, be sure to remove
public
        access when you're done.

        :param vpc: The VPC used by this example.
        :param port: The port to verify.
        :param ip_address: This computer's IP address.
        :return: The default security group of the specific VPC, and a value that
indicates
                 whether the specified port is open.
        """
        try:
            response = self.ec2_client.describe_security_groups(
                Filters=[
                    {"Name": "group-name", "Values": ["default"]},
                    {"Name": "vpc-id", "Values": [vpc["VpcId"]]},
                ]
            )
            sec_group = response["SecurityGroups"][0]
            port_is_open = False
            log.info("Found default security group %s.", sec_group["GroupId"])
            for ip_perm in sec_group["IpPermissions"]:
                if ip_perm.get("FromPort", 0) == port:
                    log.info("Found inbound rule: %s", ip_perm)
                    for ip_range in ip_perm["IpRanges"]:
                        cidr = ip_range.get("CidrIp", "")
                        if cidr.startswith(ip_address) or cidr == "0.0.0.0/0":
                            port_is_open = True
                    if ip_perm["PrefixListIds"]:
                        port_is_open = True
```

```
                    if not port_is_open:
                        log.info(
                            "The inbound rule does not appear to be open to
either this computer's IP\n"
                            "address of %s, to all IP addresses (0.0.0.0/0), or
to a prefix list ID.",
                            ip_address,
                        )
                    else:
                        break
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't verify inbound rule for port {port} for VPC
{vpc['VpcId']}: {err}"
            )
        else:
            return sec_group, port_is_open


    def open_inbound_port(self, sec_group_id, port, ip_address):
        """
        Add an ingress rule to the specified security group that allows access on
the
        specified port from the specified IP address.

        :param sec_group_id: The ID of the security group to modify.
        :param port: The port to open.
        :param ip_address: The IP address that is granted access.
        """
        try:
            self.ec2_client.authorize_security_group_ingress(
                GroupId=sec_group_id,
                CidrIp=f"{ip_address}/32",
                FromPort=port,
                ToPort=port,
                IpProtocol="tcp",
            )
            log.info(
                "Authorized ingress to %s on port %s from %s.",
                sec_group_id,
                port,
                ip_address,
            )
        except ClientError as err:
```

```
            raise AutoScalerError(
                f"Couldn't authorize ingress to {sec_group_id} on port {port}
 from {ip_address}: {err}"
            )


    def get_subnets(self, vpc_id, zones):
        """
        Gets the default subnets in a VPC for a specified list of Availability
 Zones.

        :param vpc_id: The ID of the VPC to look up.
        :param zones: The list of Availability Zones to look up.
        :return: The list of subnets found.
        """
        try:
            response = self.ec2_client.describe_subnets(
                Filters=[
                    {"Name": "vpc-id", "Values": [vpc_id]},
                    {"Name": "availability-zone", "Values": zones},
                    {"Name": "default-for-az", "Values": ["true"]},
                ]
            )
            subnets = response["Subnets"]
            log.info("Found %s subnets for the specified zones.", len(subnets))
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get subnets: {err}")
        else:
            return subnets
```

Créez une classe qui englobe les actions Elastic Load Balancing.

```
class LoadBalancer:
    """Encapsulates Elastic Load Balancing (ELB) actions."""

    def __init__(self, target_group_name, load_balancer_name, elb_client):
        """
        :param target_group_name: The name of the target group associated with
 the load balancer.
```

```
        :param load_balancer_name: The name of the load balancer.
        :param elb_client: A Boto3 Elastic Load Balancing client.
        """
        self.target_group_name = target_group_name
        self.load_balancer_name = load_balancer_name
        self.elb_client = elb_client
        self._endpoint = None


    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from a Boto3 client.

        :param resource_prefix: The prefix to give to AWS resources created by
this class.
        """
        elb_client = boto3.client("elbv2")
        return cls(f"{resource_prefix}-tg", f"{resource_prefix}-lb", elb_client)

    def endpoint(self):
        """
        Gets the HTTP endpoint of the load balancer.

        :return: The endpoint.
        """
        if self._endpoint is None:
            try:
                response = self.elb_client.describe_load_balancers(
                    Names=[self.load_balancer_name]
                )
                self._endpoint = response["LoadBalancers"][0]["DNSName"]
            except ClientError as err:
                raise LoadBalancerError(
                    f"Couldn't get the endpoint for load balancer
{self.load_balancer_name}: {err}"
                )
        return self._endpoint


    def create_target_group(self, protocol, port, vpc_id):
        """
        Creates an Elastic Load Balancing target group. The target group
specifies how
```

```
        the load balancer forward requests to instances in the group and how
instance
        health is checked.

        To speed up this demo, the health check is configured with shortened
times and
        lower thresholds. In production, you might want to decrease the
sensitivity of
        your health checks to avoid unwanted failures.

        :param protocol: The protocol to use to forward requests, such as 'HTTP'.
        :param port: The port to use to forward requests, such as 80.
        :param vpc_id: The ID of the VPC in which the load balancer exists.
        :return: Data about the newly created target group.
        """
        try:
            response = self.elb_client.create_target_group(
                Name=self.target_group_name,
                Protocol=protocol,
                Port=port,
                HealthCheckPath="/healthcheck",
                HealthCheckIntervalSeconds=10,
                HealthCheckTimeoutSeconds=5,
                HealthyThresholdCount=2,
                UnhealthyThresholdCount=2,
                VpcId=vpc_id,
            )
            target_group = response["TargetGroups"][0]
            log.info("Created load balancing target group %s.",
self.target_group_name)
        except ClientError as err:
            raise LoadBalancerError(
                f"Couldn't create load balancing target group
{self.target_group_name}: {err}"
            )
        else:
            return target_group


    def delete_target_group(self):
        """
        Deletes the target group.
        """
        done = False
```

```
        while not done:
            try:
                response = self.elb_client.describe_target_groups(
                    Names=[self.target_group_name]
                )
                tg_arn = response["TargetGroups"][0]["TargetGroupArn"]
                self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
                log.info(
                    "Deleted load balancing target group %s.",
self.target_group_name
                )
                done = True
            except ClientError as err:
                if err.response["Error"]["Code"] == "TargetGroupNotFound":
                    log.info(
                        "Load balancer target group %s not found, nothing to
do.",
                        self.target_group_name,
                    )
                    done = True
                elif err.response["Error"]["Code"] == "ResourceInUse":
                    log.info(
                        "Target group not yet released from load balancer,
waiting..."
                    )
                    time.sleep(10)
                else:
                    raise LoadBalancerError(
                        f"Couldn't delete load balancing target group
{self.target_group_name}: {err}"
                    )


    def create_load_balancer(self, subnet_ids, target_group):
        """
        Creates an Elastic Load Balancing load balancer that uses the specified
subnets
        and forwards requests to the specified target group.

        :param subnet_ids: A list of subnets to associate with the load balancer.
        :param target_group: An existing target group that is added as a listener
to the
                             load balancer.
        :return: Data about the newly created load balancer.
```

```
        """
        try:
            response = self.elb_client.create_load_balancer(
                Name=self.load_balancer_name, Subnets=subnet_ids
            )
            load_balancer = response["LoadBalancers"][0]
            log.info("Created load balancer %s.", self.load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancer_available")
            log.info("Waiting for load balancer to be available...")
            waiter.wait(Names=[self.load_balancer_name])
            log.info("Load balancer is available!")
            self.elb_client.create_listener(
                LoadBalancerArn=load_balancer["LoadBalancerArn"],
                Protocol=target_group["Protocol"],
                Port=target_group["Port"],
                DefaultActions=[
                    {
                        "Type": "forward",
                        "TargetGroupArn": target_group["TargetGroupArn"],
                    }
                ],
            )
            log.info(
                "Created listener to forward traffic from load balancer %s to
target group %s.",
                self.load_balancer_name,
                target_group["TargetGroupName"],
            )
        except ClientError as err:
            raise LoadBalancerError(
                f"Failed to create load balancer {self.load_balancer_name}"
                f"and add a listener for target group
{target_group['TargetGroupName']}: {err}"
            )
        else:
            self._endpoint = load_balancer["DNSName"]
            return load_balancer


    def delete_load_balancer(self):
        """
        Deletes a load balancer.
        """
        try:
```

```
            response = self.elb_client.describe_load_balancers(
                Names=[self.load_balancer_name]
            )
            lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
            self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
            log.info("Deleted load balancer %s.", self.load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancers_deleted")
            log.info("Waiting for load balancer to be deleted...")
            waiter.wait(Names=[self.load_balancer_name])
        except ClientError as err:
            if err.response["Error"]["Code"] == "LoadBalancerNotFound":
                log.info(
                    "Load balancer %s does not exist, nothing to do.",
                    self.load_balancer_name,
                )
            else:
                raise LoadBalancerError(
                    f"Couldn't delete load balancer {self.load_balancer_name}:
{err}"
                )


    def verify_load_balancer_endpoint(self):
        """
        Verify this computer can successfully send a GET request to the load
balancer endpoint.
        """
        success = False
        retries = 3
        while not success and retries > 0:
            try:
                lb_response = requests.get(f"http://{self.endpoint()}")
                log.info(
                    "Got response %s from load balancer endpoint.",
                    lb_response.status_code,
                )
                if lb_response.status_code == 200:
                    success = True
                else:
                    retries = 0
            except requests.exceptions.ConnectionError:
                log.info(
                    "Got connection error from load balancer endpoint,
retrying..."
```

```
                )
                retries -= 1
                time.sleep(10)
        return success

    def check_target_health(self):
        """
        Checks the health of the instances in the target group.

        :return: The health status of the target group.
        """
        try:
            tg_response = self.elb_client.describe_target_groups(
                Names=[self.target_group_name]
            )
            health_response = self.elb_client.describe_target_health(
                TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
            )
        except ClientError as err:
            raise LoadBalancerError(
                f"Couldn't check health of {self.target_group_name} targets:
 {err}"
            )
        else:
            return health_response["TargetHealthDescriptions"]
```

Créez une classe qui utilise DynamoDB pour simuler un service de recommandation.

```
class RecommendationService:
    """
    Encapsulates a DynamoDB table to use as a service that recommends books,
 movies,
    and songs.
    """

    def __init__(self, table_name, dynamodb_client):
        """
        :param table_name: The name of the DynamoDB recommendations table.
        :param dynamodb_client: A Boto3 DynamoDB client.
```

```
        """
        self.table_name = table_name
        self.dynamodb_client = dynamodb_client

    @classmethod
    def from_client(cls, table_name):
        """
        Creates this class from a Boto3 client.

        :param table_name: The name of the DynamoDB recommendations table.
        """
        ddb_client = boto3.client("dynamodb")
        return cls(table_name, ddb_client)

    def create(self):
        """
        Creates a DynamoDB table to use a recommendation service. The table has a
        hash key named 'MediaType' that defines the type of media recommended,
such as
        Book or Movie, and a range key named 'ItemId' that, combined with the
MediaType,
        forms a unique identifier for the recommended item.

        :return: Data about the newly created table.
        """
        try:
            response = self.dynamodb_client.create_table(
                TableName=self.table_name,
                AttributeDefinitions=[
                    {"AttributeName": "MediaType", "AttributeType": "S"},
                    {"AttributeName": "ItemId", "AttributeType": "N"},
                ],
                KeySchema=[
                    {"AttributeName": "MediaType", "KeyType": "HASH"},
                    {"AttributeName": "ItemId", "KeyType": "RANGE"},
                ],
                ProvisionedThroughput={"ReadCapacityUnits": 5,
"WriteCapacityUnits": 5},
            )
            log.info("Creating table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s created.", self.table_name)
        except ClientError as err:
```

```
            if err.response["Error"]["Code"] == "ResourceInUseException":
                log.info("Table %s exists, nothing to be do.", self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when creating table: {err}."
                )
        else:
            return response

    def populate(self, data_file):
        """
        Populates the recommendations table from a JSON file.

        :param data_file: The path to the data file.
        """
        try:
            with open(data_file) as data:
                items = json.load(data)
            batch = [{"PutRequest": {"Item": item}} for item in items]
            self.dynamodb_client.batch_write_item(RequestItems={self.table_name:
batch})
            log.info(
                "Populated table %s with items from %s.", self.table_name,
data_file
            )
        except ClientError as err:
            raise RecommendationServiceError(
                self.table_name, f"Couldn't populate table from {data_file}:
{err}"
            )

    def destroy(self):
        """
        Deletes the recommendations table.
        """
        try:
            self.dynamodb_client.delete_table(TableName=self.table_name)
            log.info("Deleting table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_not_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s deleted.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
```

```
                log.info("Table %s does not exist, nothing to do.",
 self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when deleting table: {err}."
                )
```

Créez une classe qui englobe les actions de Systems Manager.

```
class ParameterHelper:
    """
    Encapsulates Systems Manager parameters. This example uses these parameters
 to drive
    the demonstration of resilient architecture, such as failure of a dependency
 or
    how the service responds to a health check.
    """

    table = "doc-example-resilient-architecture-table"
    failure_response = "doc-example-resilient-architecture-failure-response"
    health_check = "doc-example-resilient-architecture-health-check"

    def __init__(self, table_name, ssm_client):
        """
        :param table_name: The name of the DynamoDB table that is used as a
 recommendation
                           service.
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.table_name = table_name

    @classmethod
    def from_client(cls, table_name):
        ssm_client = boto3.client("ssm")
        return cls(table_name, ssm_client)

    def reset(self):
        """
        Resets the Systems Manager parameters to starting values for the demo.
```

```
        These are the name of the DynamoDB recommendation table, no response when
a
        dependency fails, and shallow health checks.
        """
        self.put(self.table, self.table_name)
        self.put(self.failure_response, "none")
        self.put(self.health_check, "shallow")

    def put(self, name, value):
        """
        Sets the value of a named Systems Manager parameter.

        :param name: The name of the parameter.
        :param value: The new value of the parameter.
        """
        try:
            self.ssm_client.put_parameter(
                Name=name, Value=value, Overwrite=True, Type="String"
            )
            log.info("Setting demo parameter %s to '%s'.", name, value)
        except ClientError as err:
            raise ParameterHelperError(
                f"Couldn't set parameter {name} to {value}: {err}"
            )
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Python (Boto3) API Reference.
  - AttachLoadBalancerTargetGroupes
  - CreateAutoScalingGroup
  - CreateInstanceProfil
  - CreateLaunchModèle
  - CreateListener
  - CreateLoadÉquilibreur
  - CreateTargetGroupe
  - DeleteAutoScalingGroup
  - DeleteInstanceProfil

- DeleteLaunchModèle
- DeleteLoadÉquilibreur
- DeleteTargetGroupe
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadÉquilibreurs
- DescribeSubnets
- DescribeTargetGroupes
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Création d'un groupe IAM et ajout d'un utilisateur au groupe à l'aide d'un SDK AWS

L’exemple de code suivant illustre comment :

- Créez un groupe et accordez-lui des autorisations d'accès complètes à Amazon S3.
- Créez un nouvel utilisateur sans autorisation pour accéder à Amazon S3.
- Ajoutez l'utilisateur au groupe et montrez qu'il dispose désormais des autorisations pour Amazon S3, puis nettoyez les ressources.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
```

```
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
```

```
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
 { GroupName = groupName });
        return response.Group;
    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }
```

```
    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
```

```
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
```

```
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
 { UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
```

```
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }
```

```
    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
 { UserName = userName });
        return response.User;
```

```
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }
```

```
    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
```

```
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
```

```
    }


    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };
```

```
        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
 policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
```

```
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
 accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}


using Microsoft.Extensions.Configuration;

namespace IAMGroups;

public class IAMGroups
{
```

```
    private static ILogger logger = null!;

    // Represents JSON code for AWS full access policy for Amazon Simple
    // Storage Service (Amazon S3).
    private const string S3FullAccessPolicyDocument = "{" +
        " \"Statement\" : [{" +
            " \"Action\" : [\"s3:*\"]," +
            " \"Effect\" : \"Allow\"," +
            " \"Resource\" : \"*\"" +
        "}]" +
    "}";


    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMGroups>();

        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        var groupUserName = configuration["GroupUserName"];
        var groupName = configuration["GroupName"];
        var groupPolicyName = configuration["GroupPolicyName"];
        var groupBucketName = configuration["GroupBucketName"];
```

```
        var wrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayGroupsOverview();
        uiWrapper.PressEnter();

        // Create an IAM group.
        uiWrapper.DisplayTitle("Create IAM group");
        Console.WriteLine("Let's begin by creating a new IAM group.");
        var group = await wrapper.CreateGroupAsync(groupName);

        // Add an inline IAM policy to the group.
        uiWrapper.DisplayTitle("Add policy to group");
        Console.WriteLine("Add an inline policy to the group that allows members
to have full access to");
        Console.WriteLine("Amazon Simple Storage Service (Amazon S3) buckets.");

        await wrapper.PutGroupPolicyAsync(group.GroupName, groupPolicyName,
S3FullAccessPolicyDocument);

        uiWrapper.PressEnter();

        // Now create a new user.
        uiWrapper.DisplayTitle("Create an IAM user");
        Console.WriteLine("Now let's create a new IAM user.");
        var groupUser = await wrapper.CreateUserAsync(groupUserName);

        // Add the new user to the group.
        uiWrapper.DisplayTitle("Add the user to the group");
        Console.WriteLine("Adding the user to the group, which will give the user
the same permissions as the group.");
        await wrapper.AddUserToGroupAsync(groupUser.UserName, group.GroupName);

        Console.WriteLine($"User, {groupUser.UserName}, has been added to the
group, {group.GroupName}.");
        uiWrapper.PressEnter();

        Console.WriteLine("Now that we have created a user, and added the user to
the group, let's create an IAM access key.");

        // Create access and secret keys for the user.
        var accessKey = await wrapper.CreateAccessKeyAsync(groupUserName);
        Console.WriteLine("Key created.");
```

```
        uiWrapper.WaitABit(15, "Waiting for the access key to be ready for
use.");

        uiWrapper.DisplayTitle("List buckets");
        Console.WriteLine("To prove that the user has access to Amazon S3, list
the S3 buckets for the account.");

        var s3Client = new AmazonS3Client(accessKey.AccessKeyId,
accessKey.SecretAccessKey);
        var stsClient = new
AmazonSecurityTokenServiceClient(accessKey.AccessKeyId,
accessKey.SecretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client, stsClient);

        var buckets = await s3Wrapper.ListMyBucketsAsync();

        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName}\tcreated on:
{bucket.CreationDate}");
            });
        }

        // Show that the user also has write access to Amazon S3 by creating
        // a new bucket.
        uiWrapper.DisplayTitle("Create a bucket");
        Console.WriteLine("Since group members have full access to Amazon S3,
let's create a bucket.");
        var success = await s3Wrapper.PutBucketAsync(groupBucketName);

        if (success)
        {
            Console.WriteLine($"Successfully created the bucket:
{groupBucketName}.");
        }

        uiWrapper.PressEnter();

        Console.WriteLine("Let's list the user's S3 buckets again to show the new
bucket.");
```

```
        buckets = await s3Wrapper.ListMyBucketsAsync();

        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName}\tcreated on:
 {bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("First delete the bucket we created.");
        await s3Wrapper.DeleteBucketAsync(groupBucketName);

        Console.WriteLine($"Now remove the user, {groupUserName}, from the group,
 {groupName}.");
        await wrapper.RemoveUserFromGroupAsync(groupUserName, groupName);

        Console.WriteLine("Delete the user's access key.");
        await wrapper.DeleteAccessKeyAsync(accessKey.AccessKeyId, groupUserName);

        // Now we can safely delete the user.
        Console.WriteLine("Now we can delete the user.");
        await wrapper.DeleteUserAsync(groupUserName);

        uiWrapper.PressEnter();

        Console.WriteLine("Now we will delete the IAM policy attached to the
 group.");
        await wrapper.DeleteGroupPolicyAsync(groupName, groupPolicyName);

        Console.WriteLine("Now we delete the IAM group.");
        await wrapper.DeleteGroupAsync(groupName);

        uiWrapper.PressEnter();

        Console.WriteLine("The IAM groups demo has completed.");

        uiWrapper.PressEnter();
    }
}
```

```
namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };
```

```
        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
{ BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
```

```
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
 { BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
 full access to Amazon S3.");
```

```
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
```

```
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK for .NET .
  - AddUserToGroup
  - AttachRolePolitique
  - CreateAccessClé
  - CreateGroup
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeleteGroup
  - DeleteGroupPolitique
  - DeleteUser
  - PutGroupPolitique
  - RemoveUserFromGroup

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Créez un utilisateur IAM et jouez un rôle dans AWS STS l'utilisation d'un SDK AWS

Les exemples de code suivants montrent comment créer un utilisateur et assumer un rôle.

> **Warning**
>
> Afin d'éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

- Créer un utilisateur sans autorisation.
- Créer un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3 pour le compte.

- Ajouter une politique pour permettre à l'utilisateur d'assumer le rôle.
- Assumez le rôle et répertorier les compartiments S3 à l'aide d'informations d'identification temporaires, puis nettoyez les ressources.

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }
```

```
    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
```

```
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
 { GroupName = groupName });
        return response.Group;
    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
```

```
        });

        return response.Policy;
    }


    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
```

```
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
```

```
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
```

```
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
 { UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

```
   /// <summary>
   /// Delete an IAM user policy.
   /// </summary>
   /// <param name="policyName">The name of the IAM policy to delete.</param>
   /// <param name="userName">The username of the IAM user.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
   {
       var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }


   /// <summary>
   /// Detach an IAM policy from an IAM role.
   /// </summary>
   /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
   /// <param name="roleName">The name of the IAM role.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
   {
       var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
       {
           PolicyArn = policyArn,
           RoleName = roleName,
       });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }


   /// <summary>
   /// Gets the IAM password policy for an AWS account.
   /// </summary>
   /// <returns>The PasswordPolicy for the AWS account.</returns>
   public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
```

```
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
```

```
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }
```

```
        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }
```

```
    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
```

```
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
```

```
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
 policyName, string policyDocument)
    {
```

```
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
 accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}
```

```
using Microsoft.Extensions.Configuration;

namespace IAMBasics;

public class IAMBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();


        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();
```

```
        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();

        // First create a user. By default, the new user has
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
              $" \"AWS\": \"{userArn}\"" +
              "}," +
              "\"Action\": \"sts:AssumeRole\"" +
          "}]" +
        "}";

        // Permissions to list all buckets.
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);
```

```
        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;

        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");

        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
```

```
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);

        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);

        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
{bucket.CreationDate}");
```

```
            });
        }

        uiWrapper.PressEnter();

        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
 complete.");
        Console.WriteLine("Please wait while we clean up the resources we
 created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();

        Console.WriteLine("All done cleaning up our resources. Thank you for your
 patience.");
    }
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
```

```
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
 { BucketName = bucketName });
```

```
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
{ BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
```

```
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
 full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
 creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();
```

```
        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
```

```
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK for .NET .
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitique

- DetachRolePolitique
- PutUserPolitique

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
  echo "This demo will create an IAM user, create an IAM role, and apply the role
 to the user."
```

```
  echo_repeat "*" 88
  echo

  echo -n "Enter a name for a new IAM user: "
  get_input
  user_name=$get_input_result

  local user_arn
  user_arn=$(iam_create_user -u "$user_name")

  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created demo IAM user named $user_name"
  else
    errecho "$user_arn"
    errecho "The user failed to create. This demo will exit."
    return 1
  fi

  local access_key_response
  access_key_response=$(iam_create_user_access_key -u "$user_name")
  # shellcheck disable=SC2181
  if [[ ${?} != 0 ]]; then
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
    return 1
  fi

  IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
  local key_name=${access_key_values[0]}
  local key_secret=${access_key_values[1]}

  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."
```

```
  local assume_role_policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi

  local policy_name
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]}"

  local policy_arn
  policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created  IAM policy named $policy_name"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name"
    return 1
  fi

  if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
    echo "Attached policy $policy_arn to role $iam_role_name"
```

```
  else
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
  fi

  local assume_role_policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"$role_arn\"}]}"

  local assume_role_policy_name
  assume_role_policy_name=$(generate_random_name "test-assume-role-")

  # shellcheck disable=SC2181
  local assume_role_policy_arn
  assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
    return 1
  fi

  echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
  sleep 10
  echo_repeat "*" 88
  echo

  echo "Try to list buckets without the new user assuming the role."
  echo_repeat "*" 88
  echo

  # Set the environment variables for the created user.
  # bashsupport disable=BP2001
  export AWS_ACCESS_KEY_ID=$key_name
  # bashsupport disable=BP2001
```

```
 export AWS_SECRET_ACCESS_KEY=$key_secret

 local buckets
 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. This should not have
happened."
 else
   errecho "Because the role with permissions has not been assumed, listing
buckets failed."
 fi

 echo
 echo_repeat "*" 88
 echo "Now assume the role $iam_role_name and list the buckets."
 echo_repeat "*" 88
 echo

 local credentials

 credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 IFS=$'\t ' read -r -a credentials <<<"$credentials"

 export AWS_ACCESS_KEY_ID=${credentials[0]}
 export AWS_SECRET_ACCESS_KEY=${credentials[1]}
 # bashsupport disable=BP2001
 export AWS_SESSION_TOKEN=${credentials[2]}
```

```
  buckets=$(s3_list_buckets)

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. Listing buckets
 succeeded because of "
    echo "the assumed role."
  else
    errecho "Failed to list buckets. This should not happen."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    export AWS_SESSION_TOKEN=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
 "$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  local result=0
  export AWS_ACCESS_KEY_ID=""
  export AWS_SECRET_ACCESS_KEY=""

  echo
  echo_repeat "*" 88
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

Les fonctions IAM utilisées dans ce scénario.

```
########################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
########################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

########################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
```

```
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
```

```
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
```

```
########################################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
```

```
  echo "  -n role_name   The name of the IAM role."
  echo "  -p policy_json -- The assume role policy document."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-role \
  --role-name "$role_name" \
  --assume-role-policy-document "$policy_document" \
  --output text \
  --query Role.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
```

```
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
```

```
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
```

```
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
```

```
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
```

```
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an WS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
```

```
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an WS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
```

```
    --role-name "$role_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an WS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
```

```
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
```

```
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
```

```
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la Référence des commandes AWS CLI .
  - AttachRolePolitique

- CreateAccessClé
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessClé
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolitique
- DetachRolePolitique
- PutUserPolitique

C++

SDK pour C++

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
```

```
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;

    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
 std::endl;
        }
```

```
        user = outcome.GetResult().GetUser();
    }

    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;

        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());
```

```
        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");
```

```
        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }

        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);
```

```
        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;

                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
```

```
        if (!listBucketsOutcome.IsSuccess()) {
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());

        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
```

```
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
 role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
 s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
 LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
 std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }

            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}
```

```
bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
    bool result = true;
    if (policy.ArnHasBeenSet()) {
        // Detach the policy from the role.
        {
            Aws::IAM::Model::DetachRolePolicyRequest request;
            request.SetPolicyArn(policy.GetArn());
            request.SetRoleName(role.GetRoleName());

            Aws::IAM::Model::DetachRolePolicyOutcome outcome =
 client.DetachRolePolicy(
                    request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error Detaching policy from roles. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully detached the policy with arn "
                          << policy.GetArn()
                          << " from role " << role.GetRoleName() << "." <<
 std::endl;
            }
        }

        // Delete the policy.
        {
            Aws::IAM::Model::DeletePolicyRequest request;
            request.WithPolicyArn(policy.GetArn());

            Aws::IAM::Model::DeletePolicyOutcome outcome =
 client.DeletePolicy(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error deleting policy. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully deleted the policy with arn "
                          << policy.GetArn() << std::endl;
            }
```

```
        }

    }

    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK for C++ .
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitique
  - DetachRolePolitique
  - PutUserPolitique

Go

Kit SDK for Go V2

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Exécutez un scénario interactif à une invite de commande.

```
// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
// 1. Create a user who has no permissions.
// 2. Create a role that grants permission to list Amazon Simple Storage Service
//    (Amazon S3) buckets for the account.
// 3. Add a policy to let the user assume the role.
```

```
// 4. Try and fail to list buckets without permissions.
// 5. Assume the role and list S3 buckets using temporary credentials.
// 6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper actions.PolicyWrapper
 roleWrapper actions.RoleWrapper
 userWrapper actions.UserWrapper
 questioner demotools.IQuestioner
 helper IScenarioHelper
 isTestRun bool
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
  helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:   sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
```

```
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run() {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
   log.Println(r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))

 user := scenario.CreateUser()
 accessKey := scenario.CreateAccessKey(user)
 role := scenario.CreateRoleAndPolicies(user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(accessKey)
 scenario.ListBucketsWithAssumedRole(noPermsConfig, role)
 scenario.Cleanup(user, role)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser() *types.User {
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
 user, err := scenario.userWrapper.GetUser(userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
```

```
  log.Printf("User %v already exists.\n", *user.UserName)
 }
 log.Println(strings.Repeat("-", 88))
 return user
}

// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(user *types.User)
 *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(*user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}

// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(user *types.User)
 *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err :=
 scenario.roleWrapper.CreateRole(scenario.helper.GetName(), *user.Arn)
 if err != nil {panic(err)}
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"}, "arn:aws:s3:::*")
 if err != nil {panic(err)}
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(*listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {panic(err)}
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
```

```
 err = scenario.userWrapper.CreateUserPolicy(*user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {panic(err)}
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(accessKey
 *types.AccessKey) *aws.Config {
  log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
  scenario.questioner.Ask("Press Enter when you're ready.")
  noPermsConfig, err := config.LoadDefaultConfig(context.TODO(),
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
 ))
  if err != nil {panic(err)}

  // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)

  s3Client := s3.NewFromConfig(noPermsConfig)
  _, err = s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
  if err != nil {
   // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
```

```
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
  }
  } else {
   log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
    "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
  return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
//
// 1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//   the user's access key credentials.
// 2. Gets temporary credentials by assuming the role that grants permission to
 list the
//    buckets.
// 3. Creates an Amazon S3 client from the temporary credentials.
// 4. Lists buckets for the account. Because the temporary credentials are
 generated by
//    assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(noPermsConfig
 *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(context.TODO(),
 &sts.AssumeRoleInput{
  RoleArn:           role.Arn,
  RoleSessionName:   aws.String("AssumeRoleExampleSession"),
  DurationSeconds:   aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
```

```
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(context.TODO(),
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
  ),
 )
 if err != nil {panic(err)}

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
 if err != nil {
  log.Println("Couldn't list buckets with assumed role credentials.")
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(user *types.User, role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
 ) {
   policies, err := scenario.roleWrapper.ListAttachedRolePolicies(*role.RoleName)
   if err != nil {panic(err)}
   for _, policy := range policies {
    err = scenario.roleWrapper.DetachRolePolicy(*role.RoleName,
 *policy.PolicyArn)
    if err != nil {panic(err)}
    err = scenario.policyWrapper.DeletePolicy(*policy.PolicyArn)
    if err != nil {panic(err)}
    log.Printf("Detached policy %v from role %v and deleted the policy.\n",
```

```
      *policy.PolicyName, *role.RoleName)
    }
    err = scenario.roleWrapper.DeleteRole(*role.RoleName)
    if err != nil {panic(err)}
    log.Printf("Deleted role %v.\n", *role.RoleName)

    userPols, err := scenario.userWrapper.ListUserPolicies(*user.UserName)
    if err != nil {panic(err)}
    for _, userPol := range userPols {
     err = scenario.userWrapper.DeleteUserPolicy(*user.UserName, userPol)
     if err != nil {panic(err)}
     log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
    }
    keys, err := scenario.userWrapper.ListAccessKeys(*user.UserName)
    if err != nil {panic(err)}
    for _, key := range keys {
     err = scenario.userWrapper.DeleteAccessKey(*user.UserName, *key.AccessKeyId)
     if err != nil {panic(err)}
     log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
 *user.UserName)
    }
    err = scenario.userWrapper.DeleteUser(*user.UserName)
    if err != nil {panic(err)}
    log.Printf("Deleted user %v.\n", *user.UserName)
    log.Println(strings.Repeat("-", 88))
 }

}
```

Définissez une structure qui encapsule les actions du compte.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}
```

```
// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy() (*types.PasswordPolicy,
 error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders() ([]types.SAMLProviderListEntry,
 error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

Définissez une structure qui encapsule les actions de la politique.

```
// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version string
 Statement []PolicyStatement
```

```
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect string
 Action []string
 Principal map[string]string `json:",omitempty"`
 Resource *string `json:",omitempty"`
}



// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(maxPolicies int32) ([]types.Policy,
 error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(context.TODO(),
 &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
```

```
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(policyName string, actions []string,
  resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(context.TODO(),
 &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(policyArn string) (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(context.TODO(), &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
```

```
 return policy, err
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(policyArn string) error {
 _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

Définissez une structure qui encapsule les actions du rôle.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(maxRoles int32) ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(context.TODO(),
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}
```

```
// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(roleName string, trustedUserArn string)
 (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action: []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(context.TODO(),
 &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(roleName string) (*types.Role, error) {
 var role *types.Role
```

```
 result, err := wrapper.IamClient.GetRole(context.TODO(),
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(serviceName string,
 description string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
```

```
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(policyArn string, roleName string)
 error {
 _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),
 &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(roleName string)
 ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(roleName string, policyArn string)
 error {
```

```
 _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
 &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(roleName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteRole(context.TODO(), &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

Définissez une structure qui encapsule les actions de l'utilisateur.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(maxUsers int32) ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(context.TODO(), &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(context.TODO(), &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
```

```
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(context.TODO(),
 &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(userName string, policyName string,
 actions []string,
  roleArn string) error {
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(roleArn),
```

```
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(context.TODO(),
 &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(userName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(userName string, policyName string)
 error {
```

```
 _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
 &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(userName string) error {
 _, err := wrapper.IamClient.DeleteUser(context.TODO(), &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(userName string)
 (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
 &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}
```

```
// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(userName string, keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
 &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
 &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Pour plus d'informations sur l'API consultez les rubriques suivantes dans la référence de l'API AWS SDK for Go .
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy

- CreateRole
- CreateUser
- DeleteAccessClé
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolitique
- DetachRolePolitique
- PutUserPolitique

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de l'utilisateur IAM.

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
```

```
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
 */

public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static String userArn;

    public static void main(String[] args) throws Exception {

        final String usage = """

                Usage:
                    <username> <policyName> <roleName> <roleSessionName>
 <bucketName>\s

                Where:
                    username - The name of the IAM user to create.\s
                    policyName - The name of the policy to create.\s
                    roleName - The name of the role to create.\s
                    roleSessionName - The name of the session required for the
 assumeRole operation.\s
                    bucketName - The name of the Amazon S3 bucket from which
 objects are read.\s
                """;

        if (args.length != 5) {
            System.out.println(usage);
            System.exit(1);
```

```
        }

        String userName = args[0];
        String policyName = args[1];
        String roleName = args[2];
        String roleSessionName = args[3];
        String bucketName = args[4];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the AWS IAM example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
```

```
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();
```

```
            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
```

```
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
```

```
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();

            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
```

```
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();

        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();

            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
```

```
            System.exit(1);
        }
    }

    public static void deleteRole(IamClient iam, String roleName, String polArn)
{

        try {
            // First the policy needs to be detached.
            DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                    .policyArn(polArn)
                    .roleName(roleName)
                    .build();

            iam.detachRolePolicy(rolePolicyRequest);

            // Delete the policy.
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(polArn)
                    .build();

            iam.deletePolicy(request);
            System.out.println("*** Successfully deleted " + polArn);

            // Delete the role.
            DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();

            iam.deleteRole(roleRequest);
            System.out.println("*** Successfully deleted " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteKey(IamClient iam, String username, String
accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
```

```
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus d'informations sur l'API consultez les rubriques suivantes dans la référence de l'API AWS SDK for Java 2.x .
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole

- DeleteUser
- DeleteUserPolitique
- DetachRolePolitique
- PutUserPolitique

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créer un utilisateur IAM et un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3. L'utilisateur n'a que le droit d'assumer le rôle. Après avoir assumé le rôle, utilisez des informations d'identification temporaires pour répertorier les compartiments pour le compte.

```
import {
  CreateUserCommand,
  GetUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import { ScenarioInput } from "@aws-doc-sdk-examples/lib/scenario/index.js";

// Set the parameters.
```

```
const iamClient = new IAMClient({});
const userName = "test_name";
const policyName = "test_policy";
const roleName = "test_role";

/**
 * Create a new IAM user. If the user already exists, give
 * the option to delete and re-create it.
 * @param {string} name
 */
export const createUser = async (name, confirmAll = false) => {
  try {
    const { User } = await iamClient.send(
      new GetUserCommand({ UserName: name }),
    );
    const input = new ScenarioInput(
      "deleteUser",
      "Do you want to delete and remake this user?",
      { type: "confirm" },
    );
    const deleteUser = await input.handle({}, { confirmAll });
    // If the user exists, and you want to delete it, delete the user
    // and then create it again.
    if (deleteUser) {
      await iamClient.send(new DeleteUserCommand({ UserName: User.UserName }));
      await iamClient.send(new CreateUserCommand({ UserName: name }));
    } else {
      console.warn(
        `${name} already exists. The scenario may not work as expected.`,
      );
      return User;
    }
  } catch (caught) {
    // If there is no user by that name, create one.
    if (caught instanceof Error && caught.name === "NoSuchEntityException") {
      const { User } = await iamClient.send(
        new CreateUserCommand({ UserName: name }),
      );
      return User;
    } else {
      throw caught;
    }
  }
};
```

```
export const main = async (confirmAll = false) => {
  // Create a user. The user has no permissions by default.
  const User = await createUser(userName, confirmAll);

  if (!User) {
    throw new Error("User not created");
  }

  // Create an access key. This key is used to authenticate the new user to
  // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
 (AWS STS).
  // It's not best practice to use access keys. For more information, see
 https://aws.amazon.com/iam/resources/best-practices/.
  const createAccessKeyResponse = await iamClient.send(
    new CreateAccessKeyCommand({ UserName: userName }),
  );

  if (
    !createAccessKeyResponse.AccessKey?.AccessKeyId ||
    !createAccessKeyResponse.AccessKey?.SecretAccessKey
  ) {
    throw new Error("Access key not created");
  }

  const {
    AccessKey: { AccessKeyId, SecretAccessKey },
  } = createAccessKeyResponse;

  let s3Client = new S3Client({
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
  // thrown while the user and access keys are still stabilizing.
  await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
    try {
      return await listBuckets(s3Client);
    } catch (err) {
      if (err instanceof Error && err.name === "InvalidAccessKeyId") {
        throw err;
```

```
      }
    }
  });

  // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
  // is thrown while the user and access keys are still stabilizing.
  const { Role } = await retry(
    {
      intervalInMs: 2000,
      maxRetries: 60,
    },
    () =>
      iamClient.send(
        new CreateRoleCommand({
          AssumeRolePolicyDocument: JSON.stringify({
            Version: "2012-10-17",
            Statement: [
              {
                Effect: "Allow",
                Principal: {
                  // Allow the previously created user to assume this role.
                  AWS: User.Arn,
                },
                Action: "sts:AssumeRole",
              },
            ],
          }),
          RoleName: roleName,
        }),
      ),
  );

  if (!Role) {
    throw new Error("Role not created");
  }

  // Create a policy that allows the user to list S3 buckets.
  const { Policy: listBucketPolicy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
```

```
            Effect: "Allow",
            Action: ["s3:ListAllMyBuckets"],
            Resource: "*",
          },
        ],
      }),
      PolicyName: policyName,
    }),
  );

  if (!listBucketPolicy) {
    throw new Error("Policy not created");
  }

  // Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
  await iamClient.send(
    new AttachRolePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
      RoleName: Role.RoleName,
    }),
  );

  // Assume the role.
  const stsClient = new STSClient({
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the assume role operation until it succeeds.
  const { Credentials } = await retry(
    { intervalInMs: 2000, maxRetries: 60 },
    () =>
      stsClient.send(
        new AssumeRoleCommand({
          RoleArn: Role.Arn,
          RoleSessionName: `iamBasicScenarioSession-${Math.floor(
            Math.random() * 1000000,
          )}`,
          DurationSeconds: 900,
        }),
      ),
  );
```

```
if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
  throw new Error("Credentials not created");
}

s3Client = new S3Client({
  credentials: {
    accessKeyId: Credentials.AccessKeyId,
    secretAccessKey: Credentials.SecretAccessKey,
    sessionToken: Credentials.SessionToken,
  },
});

// List the S3 buckets again.
// Retry the list buckets operation until it succeeds. AccessDenied might
// be thrown while the role policy is still stabilizing.
await retry({ intervalInMs: 2000, maxRetries: 60 }, () =>
  listBuckets(s3Client),
);

// Clean up.
await iamClient.send(
  new DetachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
);

await iamClient.send(
  new DeletePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
  }),
);

await iamClient.send(
  new DeleteRoleCommand({
    RoleName: Role.RoleName,
  }),
);

await iamClient.send(
  new DeleteAccessKeyCommand({
    UserName: userName,
    AccessKeyId,
```

```
    }),
  );

  await iamClient.send(
    new DeleteUserCommand({
      UserName: userName,
    }),
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }

  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- Pour plus d'informations sur l'API consultez les rubriques suivantes dans la référence de l'API AWS SDK for JavaScript .
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitique
  - DetachRolePolitique

- PutUserPolitique

Kotlin

SDK pour Kotlin

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de l'utilisateur IAM.

```
suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
 <bucketName>

    Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
 operation.
        fileLocation - The file location to the JSON required to create the role
 (see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
 read.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val userName = args[0]
    val policyName = args[1]
    val roleName = args[2]
    val roleSessionName = args[3]
    val fileLocation = args[4]
```

```
    val bucketName = args[5]

    createUser(userName)
    println("$userName was successfully created.")

    val polArn = createPolicy(policyName)
    println("The policy $polArn was successfully created.")

    val roleArn = createRole(roleName, fileLocation)
    println("$roleArn was successfully created.")
    attachRolePolicy(roleName, polArn)

    println("*** Wait for 1 MIN so the resource is available.")
    delay(60000)
    assumeGivenRole(roleArn, roleSessionName, bucketName)

    println("*** Getting ready to delete the AWS resources.")
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}

suspend fun createUser(usernameVal: String?): String? {
    val request =
        CreateUserRequest {
            userName = usernameVal
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {
    val policyDocumentValue: String =
        "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
```

```
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}"

    val request =
        CreatePolicyRequest {
            policyName = policyNameVal
            policyDocument = policyDocumentValue
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}

suspend fun createRole(
    rolenameVal: String?,
    fileLocation: String?
): String? {
    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request =
        CreateRoleRequest {
            roleName = rolenameVal
            assumeRolePolicyDocument = jsonObject.toJSONString()
            description = "Created using the AWS SDK for Kotlin"
        }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(
    roleNameVal: String,
    policyArnVal: String
) {
    val request =
        ListAttachedRolePoliciesRequest {
            roleName = roleNameVal
        }
```

```
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1) {
                return
            }
        }

        val policyRequest =
            AttachRolePolicyRequest {
                roleName = roleNameVal
                policyArn = policyArnVal
            }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(
    attachedPolicies: List<AttachedPolicy>,
    policyArnVal: String
): Int {
    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(
    roleArnVal: String?,
    roleSessionNameVal: String?,
    bucketName: String
```

```
) {
    val stsClient =
        StsClient {
            region = "us-east-1"
        }

    val roleRequest =
        AssumeRoleRequest {
            roleArn = roleArnVal
            roleSessionName = roleSessionNameVal
        }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials =
        StaticCredentialsProvider {
            accessKeyId = key
            secretAccessKey = secKey
            sessionToken = secToken
        }

    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 =
        S3Client {
            credentialsProvider = staticCredentials
            region = "us-east-1"
        }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects =
        ListObjectsRequest {
            bucket = bucketName
        }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
```

```
    }
}

suspend fun deleteRole(
    roleNameVal: String,
    polArn: String
) {
    val iam = IamClient { region = "AWS_GLOBAL" }

    // First the policy needs to be detached.
    val rolePolicyRequest =
        DetachRolePolicyRequest {
            policyArn = polArn
            roleName = roleNameVal
        }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
    val request =
        DeletePolicyRequest {
            policyArn = polArn
        }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")

    // Delete the role.
    val roleRequest =
        DeleteRoleRequest {
            roleName = roleNameVal
        }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient { region = "AWS_GLOBAL" }
    val request =
        DeleteUserRequest {
            userName = userNameVal
        }
```

```
    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}

@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
    return jsonParser.parse(reader)
}
```

- Pour plus d'informations sur l'API consultez les rubriques suivantes dans la AWS Référence de l'API de SDK pour Kotlin.
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitique
  - DetachRolePolitique
  - PutUserPolitique

PHP

Kit SDK pour PHP

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
namespace Iam\Basics;

require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
use Iam\IAMService;

echo("\n");
echo("---------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("---------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
```

```
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
```

```
    echo "this should now not fail\n";
}

$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans la référence de l'API AWS SDK for PHP .
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitique
  - DetachRolePolitique
  - PutUserPolitique

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créer un utilisateur IAM et un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3. L'utilisateur n'a que le droit d'assumer le rôle. Après avoir assumé le rôle, utilisez des informations d'identification temporaires pour répertorier les compartiments pour le compte.

```
import json
import sys
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.
```

```
    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
resource
                         that has permissions to create users, roles, and
policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
                    ],
                }
```

```
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
            f"Couldn't create a policy and attach it to role {role.name}. Here's
why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
```

```
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
 "
            f"the role."
        )
    except ClientError as error:
        print(
            f"Couldn't create an inline policy for user {user.name}. Here's why:
 "
            f"{error.response['Error']['Message']}"
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
        "s3", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
```

```
            raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
        except ClientError as error:
            if error.response["Error"]["Code"] == "AccessDenied":
                print("Attempt to list buckets with no permissions: AccessDenied.")
            else:
                raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
 account.
    Uses the temporary credentials from the role to list the buckets that are
 owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the
 role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary
 credentials.
    s3_resource = boto3.resource(
        "s3",
```

```
            aws_access_key_id=temp_credentials["AccessKeyId"],
            aws_secret_access_key=temp_credentials["SecretAccessKey"],
            aws_session_token=temp_credentials["SessionToken"],
        )
        print(f"Listing buckets for the assumed role's account:")
        try:
            for bucket in s3_resource.buckets.all():
                print(bucket.name)
        except ClientError as error:
            print(
                f"Couldn't list buckets for the account. Here's why: "
                f"{error.response['Error']['Message']}"
            )
            raise




def teardown(user, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        for user_pol in user.policies.all():
            user_pol.delete()
            print("Deleted inline user policy.")
```

```
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )


def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Python (Boto3) API Reference.
  - AttachRolePolitique
  - CreateAccessClé

- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessClé
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolitique
- DetachRolePolitique
- PutUserPolitique

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créer un utilisateur IAM et un rôle qui accorde l'autorisation de répertorier les compartiments Amazon S3. L'utilisateur n'a que le droit d'assumer le rôle. Après avoir assumé le rôle, utilisez des informations d'identification temporaires pour répertorier les compartiments pour le compte.

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Waits for the specified number of seconds.
```

```
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts("Give AWS time to propagate resources...")
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Tried and failed to create demo user.")
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
```

```
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: "2012-10-17",
      Statement: [{
                   Effect: "Allow",
                   Principal: {'AWS': user.arn},
                   Action: "sts:AssumeRole"
                 }]
    }.to_json
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a role for the demo. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account,
and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
      Version: "2012-10-17",
      Statement: [{
                   Effect: "Allow",
                   Action: "s3:ListAllMyBuckets",
                   Resource: "arn:aws:s3:::*"
                 }]
    }.to_json
    policy = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
```

```
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param user [Aws::IAM::User] The user that owns the policy.
  # @param role [Aws::IAM::Role] The role that can be assumed.
  # @return [Aws::IAM::UserPolicy] The newly created policy.
  def create_user_policy(policy_name, user, role)
    policy_document = {
      Version: "2012-10-17",
      Statement: [{
                    Effect: "Allow",
                    Action: "sts:AssumeRole",
                    Resource: role.arn
                  }]
    }.to_json
    @iam_client.put_user_policy(
      user_name: user.user_name,
      policy_name: policy_name,
      policy_document: policy_document
    )
    puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an Amazon S3 resource with specified credentials. This is separated
into a
  # factory function so that it can be mocked for unit testing.
```

```
 #
 # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
 def create_s3_resource(credentials)
   Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
 end

 # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
 # Because the resource uses credentials with limited access, it may not be able
to
 # list the S3 buckets.
 #
 # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
 def list_buckets(s3_resource)
   count = 10
   s3_resource.buckets.each do |bucket|
     @logger.info "\t#{bucket.name}"
     count -= 1
     break if count.zero?
   end
 rescue Aws::Errors::ServiceError => e
   if e.code == "AccessDenied"
     puts("Attempt to list buckets with no permissions: AccessDenied.")
   else
     @logger.info("Couldn't list buckets for the account. Here's why: ")
     @logger.info("\t#{e.code}: #{e.message}")
     raise
   end
 end

 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
```

```
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: "create-use-assume-role-scenario"
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end

 # Deletes a role. If the role has policies attached, they are detached and
 # deleted before the role is deleted.
 #
 # @param role_name [String] The name of the role to delete.
 def delete_role(role_name)
   @iam_client.list_attached_role_policies(role_name:
role_name).attached_policies.each do |policy|
     @iam_client.detach_role_policy(role_name: role_name, policy_arn:
policy.policy_arn)
     @iam_client.delete_policy(policy_arn: policy.policy_arn)
     @logger.info("Detached and deleted policy #{policy.policy_name}.")
   end
   @iam_client.delete_role({ role_name: role_name })
   @logger.info("Role deleted: #{role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
why:")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
 end

 # Deletes a user. If the user has inline policies or access keys, they are
deleted
 # before the user is deleted.
 #
 # @param user [Aws::IAM::User] The user to delete.
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
```

```
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
  puts("-" * 88)
  puts("Welcome to the IAM create a user and assume a role demo!")
  puts("-" * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts("Try to list buckets with credentials for a user who has no permissions.")
  puts("Expect AccessDenied from this call.")
  scenario.list_buckets(
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key)))
  puts("Now, assume the role that grants permission.")
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key))
  puts("Here are your buckets:")
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts("Thanks for watching!")
```

```
  puts("-" * 88)
rescue Aws::Errors::ServiceError => e
  puts("Something went wrong with the demo.")
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- Pour plus d'informations sur l'API consultez les rubriques suivantes dans la référence de l'API AWS SDK for Ruby .
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitique
  - DetachRolePolitique
  - PutUserPolitique

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;

    if let Err(e) = run_iam_operations(
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
```

```
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;

    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
```

```
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
    );

    let attach_role_policy_result =
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
    match fail_client.list_buckets().send().await {
        Ok(e) => {
```

```
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
    let assumed_role = sts_client
        .assume_role()
        .role_arn(assume_role_role.arn())
        .role_session_name(&format!("{}{}", "iam_demo_assumerole_session_",
uuid))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .access_key_id(),
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
```

```
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
            println!("This should now run successfully.")
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- Pour plus d’informations sur l’API, consultez les rubriques suivantes dans AWS SDK for Rust API reference.
  - AttachRolePolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolitique
  - DetachRolePolitique
  - PutUserPolitique

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Créez des utilisateurs IAM en lecture seule et en lecture-écriture à l'aide d'un SDK AWS

L'exemple de code suivant montre comment créer des utilisateurs et leur associer des politiques.

> **Warning**
>
> Afin d’éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

- Créez des utilisateurs IAM.
- Attachez une politique pour un utilisateur afin d’obtenir et de placer des objets dans un compartiment Amazon S3.

- Attachez une politique pour le second utilisateur afin d'obtenir des objets depuis le compartiment.
- Obtenez des autorisations différentes du compartiment en fonction des informations d'identification de l'utilisateur.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de l'utilisateur IAM.

```
import logging
import time

import boto3
from botocore.exceptions import ClientError

import access_key_wrapper
import policy_wrapper

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
        raise
```

```
    else:
        return user



def update_user(user_name, new_user_name):
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user



def list_users():
    """
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users



def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
```

```
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
        raise



def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
 user_name)
        raise



def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
```

```
        raise
```

Créez des fonctions qui encapsulent les actions de politique IAM.

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::my-bucket/*' to allow actions on all
 objects
                         in the bucket named 'my-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
```

```
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy



def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

Créez des fonctions qui encapsulent les actions de clé d'accès IAM.

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.
```

```
    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair



def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

Utilisez les fonctions d'encapsulation pour créer des utilisateurs avec des politiques différentes et utilisez leurs informations d'identification pour accéder à un compartiment Amazon S3.

```
def usage_demo():
    """
    Shows how to manage users, keys, and policies.
```

```
    This demonstration creates two users: one user who can put and get objects in
an
    Amazon S3 bucket, and another user who can only get objects from the bucket.
    The demo then shows how the users can perform only the actions they are
permitted
    to perform.
    """
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management user demo.")
    print("-" * 88)
    print(
        "Users can have policies and roles attached to grant them specific "
        "permissions."
    )
    s3 = boto3.resource("s3")
    bucket = s3.create_bucket(
        Bucket=f"demo-iam-bucket-{time.time_ns()}",
        CreateBucketConfiguration={
            "LocationConstraint": s3.meta.client.meta.region_name
        },
    )
    print(f"Created an Amazon S3 bucket named {bucket.name}.")
    user_read_writer = create_user("demo-iam-read-writer")
    user_reader = create_user("demo-iam-reader")
    print(f"Created two IAM users: {user_read_writer.name} and
{user_reader.name}")
    update_user(user_read_writer.name, "demo-iam-creator")
    update_user(user_reader.name, "demo-iam-getter")
    users = list_users()
    user_read_writer = next(
        user for user in users if user.user_id == user_read_writer.user_id
    )
    user_reader = next(user for user in users if user.user_id ==
user_reader.user_id)
    print(
        f"Changed the names of the users to {user_read_writer.name} "
        f"and {user_reader.name}."
    )

    read_write_policy = policy_wrapper.create_policy(
        "demo-iam-read-write-policy",
        "Grants rights to create and get an object in the demo bucket.",
        ["s3:PutObject", "s3:GetObject"],
```

```
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(
        f"Created policy {read_write_policy.policy_name} with ARN:
{read_write_policy.arn}"
    )
    print(read_write_policy.description)
    read_policy = policy_wrapper.create_policy(
        "demo-iam-read-policy",
        "Grants rights to get an object from the demo bucket.",
        "s3:GetObject",
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(f"Created policy {read_policy.policy_name} with ARN:
{read_policy.arn}")
    print(read_policy.description)
    attach_policy(user_read_writer.name, read_write_policy.arn)
    print(f"Attached {read_write_policy.policy_name} to
{user_read_writer.name}.")
    attach_policy(user_reader.name, read_policy.arn)
    print(f"Attached {read_policy.policy_name} to {user_reader.name}.")

    user_read_writer_key = access_key_wrapper.create_key(user_read_writer.name)
    print(f"Created access key pair for {user_read_writer.name}.")
    user_reader_key = access_key_wrapper.create_key(user_reader.name)
    print(f"Created access key pair for {user_reader.name}.")

    s3_read_writer_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_read_writer_key.id,
        aws_secret_access_key=user_read_writer_key.secret,
    )
    demo_object_key = f"object-{time.time_ns()}"
    demo_object = None
    while demo_object is None:
        try:
            demo_object = s3_read_writer_resource.Bucket(bucket.name).put_object(
                Key=demo_object_key, Body=b"AWS IAM demo object content!"
            )
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
```

```
                raise
    print(
        f"Put {demo_object_key} into {bucket.name} using "
        f"{user_read_writer.name}'s credentials."
    )

    read_writer_object = s3_read_writer_resource.Bucket(bucket.name).Object(
        demo_object_key
    )
    read_writer_content = read_writer_object.get()["Body"].read()
    print(f"Got object {read_writer_object.key} using read-writer user's
credentials.")
    print(f"Object content: {read_writer_content}")

    s3_reader_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_reader_key.id,
        aws_secret_access_key=user_reader_key.secret,
    )
    demo_content = None
    while demo_content is None:
        try:
            demo_object =
s3_reader_resource.Bucket(bucket.name).Object(demo_object_key)
            demo_content = demo_object.get()["Body"].read()
            print(f"Got object {demo_object.key} using reader user's
credentials.")
            print(f"Object content: {demo_content}")
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
                raise

    try:
        demo_object.delete()
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("-" * 88)
            print(
                "Tried to delete the object using the reader user's credentials.
"
                "Got expected AccessDenied error because the reader is not "
```

```
                "allowed to delete objects."
            )
            print("-" * 88)

    access_key_wrapper.delete_key(user_reader.name, user_reader_key.id)
    detach_policy(user_reader.name, read_policy.arn)
    policy_wrapper.delete_policy(read_policy.arn)
    delete_user(user_reader.name)
    print(f"Deleted keys, detached and deleted policy, and deleted
{user_reader.name}.")

    access_key_wrapper.delete_key(user_read_writer.name, user_read_writer_key.id)
    detach_policy(user_read_writer.name, read_write_policy.arn)
    policy_wrapper.delete_policy(read_write_policy.arn)
    delete_user(user_read_writer.name)
    print(
        f"Deleted keys, detached and deleted policy, and deleted
{user_read_writer.name}."
    )

    bucket.objects.delete()
    bucket.delete()
    print(f"Emptied and deleted {bucket.name}.")
    print("Thanks for watching!")
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Python (Boto3) API Reference.
  - AttachUserPolitique
  - CreateAccessClé
  - CreatePolicy
  - CreateUser
  - DeleteAccessClé
  - DeletePolicy
  - DeleteUser
  - DetachUserPolitique
  - ListUsers

- UpdateUser

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Gérer les clés d'accès IAM à l'aide d'un SDK AWS

L'exemple de code suivant montre comment gérer les clés d'accès.

> **Warning**
>
> Afin d'éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

- Créez et répertoriez les clés d'accès.
- Découvrez quand et comment une clé d'accès a été utilisée pour la dernière fois.
- Mettez à jour et supprimez les clés d'accès.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de clé d'accès IAM.

```
import logging
import boto3
from botocore.exceptions import ClientError
```

```
logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys



def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair
```

```
def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.

    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response



def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
    :param activate: When True, the key is activated. Otherwise, the key is
 deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
        else:
            key.deactivate()
```

```
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
 key_id)
    except ClientError:
        logger.exception(
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
 key_id
        )
        raise



def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

Utilisez les fonctions d'enveloppe pour effectuer des actions de clé d'accès pour l'utilisateur actuel.

```
def usage_demo():
    """Shows how to create and manage access keys."""

    def print_keys():
        """Gets and prints the current keys for a user."""
        current_keys = list_keys(current_user_name)
        print("The current user's keys are now:")
        print(*[f"{key.id}: {key.status}" for key in current_keys], sep="\n")

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
```

```
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management access key demo.")
    print("-" * 88)
    current_user_name = iam.CurrentUser().user_name
    print(
        f"This demo creates an access key for the current user "
        f"({current_user_name}), manipulates the key in a few ways, and then "
        f"deletes it."
    )
    all_keys = list_keys(current_user_name)
    if len(all_keys) == 2:
        print(
            "The current user already has the maximum of 2 access keys. To run "
            "this demo, either delete one of the access keys or use a user "
            "that has only 1 access key."
        )
    else:
        new_key = create_key(current_user_name)
        print(f"Created a new key with id {new_key.id} and secret
{new_key.secret}.")
        print_keys()
        existing_key = next(key for key in all_keys if key != new_key)
        last_use = get_last_use(existing_key.id)["AccessKeyLastUsed"]
        print(
            f"Key {all_keys[0].id} was last used to access
{last_use['ServiceName']} "
            f"on {last_use['LastUsedDate']}"
        )
        update_key(current_user_name, new_key.id, False)
        print(f"Key {new_key.id} is now deactivated.")
        print_keys()
        delete_key(current_user_name, new_key.id)
        print_keys()
        print("Thanks for watching!")
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Python (Boto3) API Reference.
  - CreateAccessClé
  - DeleteAccessClé

- GetAccessKeyLastUsagé
- ListAccessClés
- UpdateAccessClé

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Gérer les politiques IAM à l'aide d'un SDK AWS

L'exemple de code suivant illustre comment :

- Créez et répertoriez des stratégies.
- Créez et obtenez des versions de stratégies.
- Rétablissez une stratégie à une version précédente.
- Supprimez des stratégies.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de politique IAM.

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError
```

```
logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::my-bucket/*' to allow actions on all
 objects
                         in the bucket named 'my-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy



def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
```

```
                    'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
        return policies



def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
```

```
            logger.exception("Couldn't create a policy version for %s.", policy_arn)
            raise
        else:
            return policy_version



def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement



def rollback_policy_version(policy_arn):
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
```

```
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
        logger.info("Set %s as the default version.",
 rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
 default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
 so "
                "nothing to roll back to.",
                default_version.version_id,
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version



def delete_policy(policy_arn):
    """
    Deletes a policy.
```

```
    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

Utilisez les fonctions d'enveloppe pour créer des politiques, mettre à jour des versions et obtenir des informations à leur sujet.

```
def usage_demo():
    """Shows how to use the policy functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management policy demo.")
    print("-" * 88)
    print(
        "Policies let you define sets of permissions that can be attached to "
        "other IAM resources, like users and roles."
    )
    bucket_arn = f"arn:aws:s3:::made-up-bucket-name"
    policy = create_policy(
        "demo-iam-policy",
        "Policy for IAM demonstration.",
        ["s3:ListObjects"],
        bucket_arn,
    )
    print(f"Created policy {policy.policy_name}.")
    policies = list_policies("Local")
    print(f"Your account has {len(policies)} managed policies:")
    print(*[pol.policy_name for pol in policies], sep=", ")
    time.sleep(1)
    policy_version = create_policy_version(
        policy.arn, ["s3:PutObject"], bucket_arn, True
    )
    print(
        f"Added policy version {policy_version.version_id} to policy "
```

```
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
    print(f"The default policy statement for {policy.policy_name} is:")
    pprint.pprint(default_statement)
    rollback_version = rollback_policy_version(policy.arn)
    print(
        f"Rolled back to version {rollback_version.version_id} for "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
    print(f"The default policy statement for {policy.policy_name} is now:")
    pprint.pprint(default_statement)
    delete_policy(policy.arn)
    print(f"Deleted policy {policy.policy_name}.")
    print("Thanks for watching!")
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Python (Boto3) API Reference.
  - CreatePolicy
  - CreatePolicyVersion
  - DeletePolicy
  - DeletePolicyVersion
  - GetPolicyVersion
  - ListPolicies
  - ListPolicyVersions
  - SetDefaultPolicyVersion

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Gérer les rôles IAM à l'aide d'un SDK AWS

L'exemple de code suivant illustre comment :

- Créez un rôle IAM.
- Attacher et détacher des stratégies à/d'un rôle
- Supprimer un rôle

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions de rôle IAM.

```
import json
import logging
import pprint

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
    """
    trust_policy = {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
```

```
            }
            for service in allowed_services
        ],
    }

    try:
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role



def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise



def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
```

```
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise



def delete_role(role_name):
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

Utilisez les fonctions d'enveloppe pour créer un rôle, puis attacher et détacher une politique.

```
def usage_demo():
    """Shows how to use the role functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management role demo.")
    print("-" * 88)
    print(
        "Roles let you define sets of permissions and can be assumed by "
        "other entities, like users and services."
    )
    print("The first 10 roles currently in your account are:")
    roles = list_roles(10)
    print(f"The inline policies for role {roles[0].name} are:")
    list_policies(roles[0].name)
```

```
    role = create_role(
        "demo-iam-role", ["lambda.amazonaws.com",
"batchoperations.s3.amazonaws.com"]
    )
    print(f"Created role {role.name}, with trust policy:")
    pprint.pprint(role.assume_role_policy_document)
    policy_arn = "arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess"
    attach_policy(role.name, policy_arn)
    print(f"Attached policy {policy_arn} to {role.name}.")
    print(f"Policies attached to role {role.name} are:")
    list_attached_policies(role.name)
    detach_policy(role.name, policy_arn)
    print(f"Detached policy {policy_arn} from {role.name}.")
    delete_role(role.name)
    print(f"Deleted {role.name}.")
    print("Thanks for watching!")
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Python (Boto3) API Reference.
  - AttachRolePolitique
  - CreateRole
  - DeleteRole
  - DetachRolePolitique

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Gérez votre compte IAM à l'aide d'un SDK AWS

L'exemple de code suivant illustre comment :

- Obtenez et mettez à jour l'alias du compte.
- Générez un rapport sur les utilisateurs et les informations d'identification.
- Obtenez un résumé de l'utilisation du compte.

- Obtenez des informations pour tous les utilisateurs, les groupes, les rôles et les politiques dans votre compte, y compris les relations entre eux.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez des fonctions qui encapsulent les actions du compte IAM.

```
import logging
import pprint
import sys
import time
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
```

```
    else:
        return response["AccountAliases"]



def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise



def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise



def generate_credential_report():
    """
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
```

```
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
 %s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
 account.")
        raise
    else:
        return response



def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]



def get_summary():
    """
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
```

```
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
        raise
    else:
        return summary.summary_map



def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

Appelez les fonctions d'encapsulation pour modifier l'alias du compte et obtenir des rapports sur le compte.

```
def usage_demo():
    """Shows how to use the account functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management account demo.")
```

```
    print("-" * 88)
    print(
        "Setting an account alias lets you use the alias in your sign-in URL "
        "instead of your account number."
    )
    old_aliases = list_aliases()
    if len(old_aliases) > 0:
        print(f"Your account currently uses '{old_aliases[0]}' as its alias.")
    else:
        print("Your account currently has no alias.")
    for index in range(1, 3):
        new_alias = f"alias-{index}-{time.time_ns()}"
        print(f"Setting your account alias to {new_alias}")
        create_alias(new_alias)
    current_aliases = list_aliases()
    print(f"Your account alias is now {current_aliases}.")
    delete_alias(current_aliases[0])
    print(f"Your account now has no alias.")
    if len(old_aliases) > 0:
        print(f"Restoring your original alias back to {old_aliases[0]}...")
        create_alias(old_aliases[0])

    print("-" * 88)
    print("You can get various reports about your account.")
    print("Let's generate a credentials report...")
    report_state = None
    while report_state != "COMPLETE":
        cred_report_response = generate_credential_report()
        old_report_state = report_state
        report_state = cred_report_response["State"]
        if report_state != old_report_state:
            print(report_state, sep="")
        else:
            print(".", sep="")
        sys.stdout.flush()
        time.sleep(1)
    print()
    cred_report = get_credential_report()
    col_count = 3
    print(f"Got credentials report. Showing only the first {col_count} columns.")
    cred_lines = [
        line.split(",")[:col_count] for line in
cred_report.decode("utf-8").split("\n")
    ]
```

```
    col_width = max([len(item) for line in cred_lines for item in line]) + 2
    for line in cred_report.decode("utf-8").split("\n"):
        print(
            "".join(element.ljust(col_width) for element in line.split(",")
[:col_count])
        )

    print("-" * 88)
    print("Let's get an account summary.")
    summary = get_summary()
    print("Here's your summary:")
    pprint.pprint(summary)

    print("-" * 88)
    print("Let's get authorization details!")
    details = get_authorization_details([])
    see_details = input("These are pretty long, do you want to see them (y/n)? ")
    if see_details.lower() == "y":
        pprint.pprint(details)

    print("-" * 88)
    pw_policy_created = None
    see_pw_policy = input("Want to see the password policy for the account (y/n)?
 ")
    if see_pw_policy.lower() == "y":
        while True:
            if print_password_policy():
                break
            else:
                answer = input(
                    "Do you want to create a default password policy (y/n)? "
                )
                if answer.lower() == "y":
                    pw_policy_created = iam.create_account_password_policy()
                else:
                    break
    if pw_policy_created is not None:
        answer = input("Do you want to delete the password policy (y/n)? ")
        if answer.lower() == "y":
            pw_policy_created.delete()
            print("Password policy deleted.")

    print("The SAML providers for your account are:")
    list_saml_providers(10)
```

```
    print("-" * 88)
    print("Thanks for watching.")
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Python (Boto3) API Reference.
  - CreateAccountAlias
  - DeleteAccountAlias
  - GenerateCredentialRapport
  - GetAccountAuthorizationDetails
  - GetAccountRésumé
  - GetCredentialRapport
  - ListAccountAlias

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Restaurer une version d'une politique IAM à l'aide d'un SDK AWS

L'exemple de code suivant illustre comment :

- Obtenez la liste des versions de stratégie par ordre chronologique.
- Trouver la version de stratégie par défaut.
- Définissez la version précédente de la stratégie comme version par défaut.
- Supprimez l'ancienne version par défaut.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
def rollback_policy_version(policy_arn):
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
```

```
        logger.info("Set %s as the default version.",
rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
so "
                "nothing to roll back to.",
                default_version.version_id,
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version
```

- Pour plus d'informations sur l'API, consultez les rubriques suivantes dans AWS SDK for Python (Boto3) API Reference.
  - DeletePolicyVersion
  - ListPolicyVersions
  - SetDefaultPolicyVersion

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utiliser l'API IAM Policy Builder à l'aide d'un SDK AWS

L'exemple de code suivant illustre comment :

- Créez des politiques IAM à l'aide de l'API orientée objet.
- Utilisez l'API IAM Policy Builder avec le service IAM.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Les exemples utilisent les importations suivantes.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.policybuilder.iam.IamConditionOperator;
import software.amazon.awssdk.policybuilder.iam.IamEffect;
import software.amazon.awssdk.policybuilder.iam.IamPolicy;
import software.amazon.awssdk.policybuilder.iam.IamPolicyWriter;
import software.amazon.awssdk.policybuilder.iam.IamPrincipal;
import software.amazon.awssdk.policybuilder.iam.IamPrincipalType;
import software.amazon.awssdk.policybuilder.iam.IamResource;
import software.amazon.awssdk.policybuilder.iam.IamStatement;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyVersionResponse;
import software.amazon.awssdk.services.sts.StsClient;

import java.net.URLDecoder;
import java.nio.charset.StandardCharsets;
import java.util.Arrays;
import java.util.List;
```

Créez une politique basée sur le temps.

```
        public String timeBasedPolicyExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")
```

```
                                                .addResource(IamResource.ALL)
                                                .addCondition(b1 -> b1

.operator(IamConditionOperator.DATE_GREATER_THAN)

.key("aws:CurrentTime")

.value("2020-04-01T00:00:00Z"))
                                                .addCondition(b1 -> b1

.operator(IamConditionOperator.DATE_LESS_THAN)

.key("aws:CurrentTime")

.value("2020-06-30T23:59:59Z")))
                                .build();

                // Use an IamPolicyWriter to write out the JSON string to a more
readable
                // format.
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true)
                                .build());
        }
```

Créez une politique comportant plusieurs conditions.

```
        public String multipleConditionsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")

.addAction("dynamodb:BatchGetItem")
                                                .addAction("dynamodb:Query")
                                                .addAction("dynamodb:PutItem")
                                                .addAction("dynamodb:UpdateItem")
                                                .addAction("dynamodb:DeleteItem")

.addAction("dynamodb:BatchWriteItem")

.addResource("arn:aws:dynamodb:*:*:table/table-name")
```

```
 .addConditions(IamConditionOperator.STRING_EQUALS

 .addPrefix("ForAllValues:"),

 "dynamodb:Attributes",
                                                                    List.of("column-
name1", "column-name2", "column-name3"))
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS

 .addSuffix("IfExists"))

 .key("dynamodb:Select")

 .value("SPECIFIC_ATTRIBUTES")))
                                .build();

                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Utilisez des principaux dans une politique.

```
        public String specifyPrincipalsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.DENY)
                                                .addAction("s3:*")
                                                .addPrincipal(IamPrincipal.ALL)

 .addResource("arn:aws:s3:::BUCKETNAME/*")

 .addResource("arn:aws:s3:::BUCKETNAME")
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.ARN_NOT_EQUALS)

 .key("aws:PrincipalArn")

 .value("arn:aws:iam::444455556666:user/user-name")))
```

```
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Autorisez l'accès intercompte .

```
        public String allowCrossAccountAccessExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)

 .addPrincipal(IamPrincipalType.AWS, "111122223333")
                                                .addAction("s3:PutObject")
                                                .addResource("arn:aws:s3:::DOC-
EXAMPLE-BUCKET/*")
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS)
                                                                .key("s3:x-amz-
acl")
                                                                .value("bucket-
owner-full-control")))
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Créez et chargez une `IamPolicy`.

```
        public String createAndUploadPolicyExample(IamClient iam, String
 accountID, String policyName) {
                // Build the policy.
                IamPolicy policy = IamPolicy.builder() // 'version' defaults to
 "2012-10-17".
                                .addStatement(IamStatement.builder()
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:PutItem")

 .addResource("arn:aws:dynamodb:us-east-1:" + accountID
```

```
                                                                    + ":table/
exampleTableName")
                                                .build())
                                .build();
                // Upload the policy.
                iam.createPolicy(r ->
 r.policyName(policyName).policyDocument(policy.toJson()));
                return
 policy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

Téléchargez et utilisez une `IamPolicy`.

```
        public String createNewBasedOnExistingPolicyExample(IamClient iam, String
 accountID, String policyName,
                        String newPolicyName) {

                String policyArn = "arn:aws:iam::" + accountID + ":policy/" +
 policyName;
                GetPolicyResponse getPolicyResponse = iam.getPolicy(r ->
 r.policyArn(policyArn));

                String policyVersion =
 getPolicyResponse.policy().defaultVersionId();
                GetPolicyVersionResponse getPolicyVersionResponse = iam
                                .getPolicyVersion(r ->
 r.policyArn(policyArn).versionId(policyVersion));

                // Create an IamPolicy instance from the JSON string returned
 from IAM.
                String decodedPolicy =
 URLDecoder.decode(getPolicyVersionResponse.policyVersion().document(),
                                StandardCharsets.UTF_8);
                IamPolicy policy = IamPolicy.fromJson(decodedPolicy);

                /*
                 * All IamPolicy components are immutable, so use the copy method
 that creates a
                 * new instance that
                 * can be altered in the same method call.
                 *
```

```
                 * Add the ability to get an item from DynamoDB as an additional
action.
                 */
                IamStatement newStatement = policy.statements().get(0).copy(s ->
s.addAction("dynamodb:GetItem"));

                // Create a new statement that replaces the original statement.
                IamPolicy newPolicy = policy.copy(p ->
p.statements(Arrays.asList(newStatement)));

                // Upload the new policy. IAM now has both policies.
                iam.createPolicy(r -> r.policyName(newPolicyName)
                                .policyDocument(newPolicy.toJson()));

                return
newPolicy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

- Pour de plus amples informations, consultez le Guide du développeur AWS SDK for Java 2.x.
- Pour plus d'informations sur l'API consultez les rubriques suivantes dans la référence de l'API AWS SDK for Java 2.x .
  - CreatePolicy
  - GetPolicy
  - GetPolicyVersion

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Exemples de code pour AWS STS l'utilisation des AWS SDK

Les exemples de code suivants montrent comment utiliser AWS STS un kit de développement AWS logiciel (SDK).

Les actions sont des extraits de code de programmes plus larges et doivent être exécutées dans leur contexte. Alors que les actions vous indiquent comment appeler des fonctions de service

individuelles, vous pouvez les voir en contexte dans leurs scénarios associés et dans des exemples interservices.

Les Scénarios sont des exemples de code qui vous montrent comment accomplir une tâche spécifique en appelant plusieurs fonctions au sein d'un même service.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes du kit de développement logiciel (SDK).

Exemples de code

- Actions relatives à AWS STS l'utilisation des AWS SDK
  - Utilisation AssumeRole avec un AWS SDK ou une CLI
  - Utilisation AssumeRoleWithWebIdentity avec un AWS SDK ou une CLI
  - Utilisation DecodeAuthorizationMessage avec un AWS SDK ou une CLI
  - Utilisation GetFederationToken avec un AWS SDK ou une CLI
  - Utilisation GetSessionToken avec un AWS SDK ou une CLI
- Scénarios d' AWS STS utilisation des AWS SDK
  - Assumez un rôle IAM qui nécessite un jeton MFA à l' AWS STS aide d'un SDK AWS
  - Création d'une URL AWS STS pour les utilisateurs fédérés à l'aide d'un SDK AWS
  - Obtenez un jeton de session qui nécessite un jeton MFA à AWS STS l'aide d'un SDK AWS

Actions relatives à AWS STS l'utilisation des AWS SDK

Les exemples de code suivants montrent comment effectuer des AWS STS actions individuelles avec AWS les SDK. Ces extraits appellent l' AWS STS API et sont des extraits de code de programmes plus volumineux qui doivent être exécutés en contexte. Chaque exemple inclut un lien vers GitHub, où vous pouvez trouver des instructions pour configurer et exécuter le code.

Les exemples suivants incluent uniquement les actions les plus couramment utilisées. Pour obtenir la liste complète, veuillez consulter la Référence d'API AWS Security Token Service (AWS STS).

Exemples

- Utilisation AssumeRole avec un AWS SDK ou une CLI
- Utilisation AssumeRoleWithWebIdentity avec un AWS SDK ou une CLI
- Utilisation DecodeAuthorizationMessage avec un AWS SDK ou une CLI

- Utilisation GetFederationToken avec un AWS SDK ou une CLI
- Utilisation GetSessionToken avec un AWS SDK ou une CLI

Utilisation **`AssumeRole`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`AssumeRole`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans les exemples de code suivants :

- Assumer un rôle IAM qui nécessite un jeton MFA
- Créer une URL pour utilisateurs fédérés

.NET

AWS SDK for .NET

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
```

```
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);

            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials);
```

```
            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK for .NET API.

Bash

AWS CLI avec le script Bash

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
#############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
```

```
}

###############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
        ech o"Invalid parameter"
        usage
        return 1
```

```
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS CLI commandes.

C++

SDK pour C++

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
                             Aws::Auth::AWSCredentials &credentials,
```

```
                                const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }

    return outcome.IsSuccess();
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK for C++ API.

CLI

AWS CLI

Pour endosser un rôle

La commande `assume-role` suivante extrait un ensemble d'informations d'identification à court terme pour le rôle IAM `s3-access-example`.

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

Sortie :

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

La sortie de la commande contient une clé d'accès, une clé secrète et un jeton de session que vous pouvez utiliser pour vous authentifier auprès d' AWS.

Pour l'utilisation de la AWS CLI, vous pouvez configurer un profil nommé associé à un rôle. Lorsque vous utilisez le profil, la AWS CLI appelle assume-role et gère les informations d'identification pour vous. Pour plus d'informations, consultez la section Utiliser un rôle IAM dans la AWS CLI du Guide de l'utilisateur de la AWS CLI.

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS CLI commandes.

Java

SDK pour Java 2.x

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version": "2012-10-17",
 * "Statement": [
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code
 * example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
```

```
 * Role" in the AWS Directory Service guide.
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, rn:aws:iam::000008047983:role/s3role).\s
                    roleSessionName - An identifier for the assumed role session
 (for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
 String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
```

```
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
 DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
 exTime);

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK for Java 2.x API.

JavaScript

SDK pour JavaScript (v3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez le client.

```
import { STSClient } from "@aws-sdk/client-sts";
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

Assumez le rôle IAM.

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
  } catch (err) {
    console.error(err);
  }
};
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK for JavaScript API.

SDK pour JavaScript (v2)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });

//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);
```

```
  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK for JavaScript API.

PowerShell

Outils pour PowerShell

Exemple 1 : renvoie un ensemble d'informations d'identification temporaires (clé d'accès, clé secrète et jeton de session) qui peuvent être utilisées pendant une heure pour accéder à AWS des ressources auxquelles l'utilisateur demandeur n'a pas normalement accès. Les informations d'identification renvoyées disposent des autorisations autorisées par la politique d'accès du rôle assumé et par la politique fournie (vous ne pouvez pas utiliser la politique fournie pour accorder des autorisations supérieures à celles définies par la politique d'accès du rôle assumé).

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -Policy "...JSON policy..." -DurationInSeconds 3600
```

Exemple 2 : renvoie un ensemble d'informations d'identification temporaires, valides pendant une heure, dotées des mêmes autorisations que celles définies dans la politique d'accès du rôle assumé.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600
```

Exemple 3 : renvoie un ensemble d'informations d'identification temporaires fournissant le numéro de série et le jeton généré à partir d'un MFA associé aux informations d'identification utilisateur utilisées pour exécuter l'applet de commande.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -SerialNumber "GAHT12345678" -TokenCode "123456"
```

Exemple 4 : renvoie un ensemble d'informations d'identification temporaires qui ont assumé un rôle défini dans un compte client. Pour chaque rôle que le tiers peut assumer, le compte client doit créer un rôle à l'aide d'un identifiant qui doit être transmis dans le ExternalId paramètre - chaque fois que le rôle est assumé.

```
Use-STSRole -RoleSessionName "Bob" -RoleArn "arn:aws:iam::123456789012:role/demo"
 -DurationInSeconds 3600 -ExternalId "ABC123"
```

- Pour plus de détails sur l'API, consultez la section AssumeRoleRéférence des AWS Tools for PowerShell applets de commande.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Assumez un rôle IAM qui nécessite un jeton MFA et utilisez des informations d'identification temporaires pour répertorier les compartiments Amazon S3 pour le compte.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
```

```
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Pour plus de détails sur l'API, consultez AssumeRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Ruby

Kit SDK pour Ruby

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: "create-use-assume-role-scenario"
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section Référence des AWS SDK for Ruby API.

Rust

SDK pour Rust

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
}
```

- Pour plus de détails sur l'API, voir AssumeRolela section de référence de l'API AWS SDK for Rust.

Swift

Kit SDK pour Swift

> **Note**
>
> Ceci est une documentation préliminaire pour une fonctionnalité en version de prévisualisation. Elle est susceptible d'être modifiée.

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

```
public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
                async throws -> STSClientTypes.Credentials {
    let input = AssumeRoleInput(
        roleArn: role.arn,
        roleSessionName: sessionName
    )
    do {
        let output = try await stsClient.assumeRole(input: input)

        guard let credentials = output.credentials else {
            throw ServiceHandlerError.authError
        }

        return credentials
    } catch {
        throw error
    }
}
```

- Pour plus de détails sur l'API, reportez-vous AssumeRoleà la section AWS SDK pour la référence de l'API Swift.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`AssumeRoleWithWebIdentity`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`AssumeRoleWithWebIdentity`.

CLI

AWS CLI

Pour obtenir des informations d'identification à court terme pour un rôle authentifié avec Web Identity (OAuth 2."0)

La commande `assume-role-with-web-identity` suivante extrait un ensemble d'informations d'identification à court terme pour le rôle IAM `app1`. La demande est authentifiée à l'aide du jeton d'identité Web fourni par le fournisseur d'identité Web spécifié. Deux politiques supplémentaires sont appliquées à la session afin de restreindre davantage ce que l'utilisateur peut faire. Les informations d'identification renvoyées expirent une heure après leur génération.

```
aws sts assume-role-with-web-identity \
    --duration-seconds 3600 \
    --role-session-name "app1" \
    --provider-id "www.amazon.com" \
    --policy-arns "arn:aws:iam::123456789012:policy/
q=webidentitydemopolicy1","arn:aws:iam::123456789012:policy/
webidentitydemopolicy2" \
    --role-arn arn:aws:iam::123456789012:role/FederatedWebIdentityRole \
    --web-identity-token "Atza
%7CIQEBLjAsAhRFiXuWpUXuRvQ9PZL3GMFcYevydwIUFAHZwXZXXXXXXXXJnrulxKDHwy87oGKPznh0D6bEQZTSCz
CrKqjG7nPBjNIL016GGvuS5gSvPRUxWES3VYfm1wl7WTI7jn-Pcb6M-
buCgHhFOzTQxod27L9CqnOLio7N3gZAGpsp6n1-
AJBOCJckcyXe2c6uD0srOJeZlKUm2eTDVMf8IehDVI0r1QOnTV6KzzAI3OY87Vd_cVMQ"
```

Sortie :

```
{
    "SubjectFromWebIdentityToken": "amzn1.account.AF6RHO7KZU5XRVQJGXK6HB56KR2A"
    "Audience": "client.5498841531868486423.1548@apps.example.com",
    "AssumedRoleUser": {
        "Arn": "arn:aws:sts::123456789012:assumed-role/FederatedWebIdentityRole/
app1",
        "AssumedRoleId": "AROACLKWSDQRAOEXAMPLE:app1"
    }
    "Credentials": {
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    },
    "Provider": "www.amazon.com"
}
```

Pour plus d’informations, consultez Demande d’informations d’identification temporaires de sécurité dans le Guide de l’utilisateur AWS IAM.

- Pour plus de détails sur l'API, consultez AssumeRoleWithWebIdentity in AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : renvoie un ensemble temporaire d'informations d'identification, valides pendant une heure, pour un utilisateur authentifié auprès du fournisseur d'identité Login with Amazon. Les informations d'identification reposent sur la politique d'accès associée au rôle identifié par l'ARN du rôle. Vous pouvez éventuellement transmettre une politique JSON au paramètre -Policy afin d'affiner davantage les autorisations d'accès (vous ne pouvez pas accorder plus d'autorisations que celles disponibles dans les autorisations associées au rôle). La valeur fournie à -WebIdentityToken est l'identifiant utilisateur unique renvoyé par le fournisseur d'identité.

```
Use-STSWebIdentityRole -DurationInSeconds 3600 -ProviderId "www.amazon.com"
 -RoleSessionName "app1" -RoleArn "arn:aws:iam::123456789012:role/
FederatedWebIdentityRole" -WebIdentityToken "Atza...DVI0r1"
```

- Pour plus de détails sur l'API, consultez AssumeRoleWithWebIdentity in AWS Tools for PowerShell Cmdlet Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`DecodeAuthorizationMessage`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`DecodeAuthorizationMessage`.

CLI

AWS CLI

Pour décoder un message d'autorisation codé renvoyé en réponse à une demande

L'`decode-authorization-message`exemple suivant décode des informations supplémentaires sur le statut d'autorisation d'une demande à partir d'un message codé renvoyé en réponse à une demande Amazon Web Services.

```
aws sts decode-authorization-message \
    --encoded-message EXAMPLEWodyRNrtlQARDip-
eTA6i6DrlUhHhPQrLWB_lAbl5pAKxl9mPDLexYcGBreyIKQC1BGBIpBKr3dFDkwqeO7e2NMk5j_hmzAiChJN-8oy3
Ojau7BMjOTWwOtHPhV_Zaz87yENdipr745EjQwRd5LaoL3vN8_5ZfA9UiBMKDgVh1gjqZJFUiQoubv78V1RbHNYnK
pOu3FZjwYStfvTb3GHs3-6rLribGO9jZOktkfE6vqxlFzLyeDr4P2ihC1wty9tArCvvGzIAUNmARQJ2VVWPxioqgo
JWP5pwe_mAyqh0NLw-r1S56YC_90onj9A80sNrHlI-
tIiNd7tgNTYzDuPQYD2FMDBnp82V9eVmYGtPp5NIeSpuf3fOHanFuBZgENxZQZ2dlH3xJGMTtYayzZrRXjiq_SfX9
FaoPIb8LmmKVBLpIB0iFhU9sEHPqKHVPi6jdxXqKaZaFGvYVmVOiuQdNQKuyk0p067POFrZECLjjOtNPBOZCcuEKE
```

Sortie :

```
{
    "DecodedMessage": "{\"allowed\":false,\"explicitDeny\":true,
\"matchedStatements\":{\"items\":[{\"statementId\":\"VisualEditor0\",\"effect
\":\"DENY\",\"principals\":{\"items\":[{\"value\":\"AROA123456789EXAMPLE
```

```
\"}]},\"principalGroups\":{\"items\":[]},\"actions\":{\"items\":[{\"value
\":\"ec2:RunInstances\"}]},\"resources\":{\"items\":[{\"value\":\"*
\"}]},\"conditions\":{\"items\":[]}}]},\"failures\":{\"items\":[]},
\"context\":{\"principal\":{\"id\":\"AROA123456789EXAMPLE:Ana\",\"arn
\":\"arn:aws:sts::111122223333:assumed-role/Developer/Ana\"},\"action\":
\"RunInstances\",\"resource\":\"arn:aws:ec2:us-east-1:111122223333:instance/*
\",\"conditions\":{\"items\":[{\"key\":\"ec2:MetadataHttpPutResponseHopLimit\",
\"values\":{\"items\":[{\"value\":\"2\"}]}},{\"key\":\"ec2:InstanceMarketType
\",\"values\":{\"items\":[{\"value\":\"on-demand\"}]}},{\"key\":\"aws:Resource
\",\"values\":{\"items\":[{\"value\":\"instance/*\"}]}},{\"key\":\"aws:Account
\",\"values\":{\"items\":[{\"value\":\"111122223333\"}]}},{\"key\":
\"ec2:AvailabilityZone\",\"values\":{\"items\":[{\"value\":\"us-east-1f\"}]}},
{\"key\":\"ec2:ebsOptimized\",\"values\":{\"items\":[{\"value\":\"false\"}]}},
{\"key\":\"ec2:IsLaunchTemplateResource\",\"values\":{\"items\":[{\"value\":
\"false\"}]}},{\"key\":\"ec2:InstanceType\",\"values\":{\"items\":[{\"value
\":\"t2.micro\"}]}},{\"key\":\"ec2:RootDeviceType\",\"values\":{\"items\":
[{\"value\":\"ebs\"}]}},{\"key\":\"aws:Region\",\"values\":{\"items\":[{\"value
\":\"us-east-1\"}]}},{\"key\":\"ec2:MetadataHttpEndpoint\",\"values\":{\"items
\":[{\"value\":\"enabled\"}]}},{\"key\":\"aws:Service\",\"values\":{\"items
\":[{\"value\":\"ec2\"}]}},{\"key\":\"ec2:InstanceID\",\"values\":{\"items\":
[{\"value\":\"*\"}]}},{\"key\":\"ec2:MetadataHttpTokens\",\"values\":{\"items
\":[{\"value\":\"required\"}]}},{\"key\":\"aws:Type\",\"values\":{\"items
\":[{\"value\":\"instance\"}]}},{\"key\":\"ec2:Tenancy\",\"values\":{\"items
\":[{\"value\":\"default\"}]}},{\"key\":\"ec2:Region\",\"values\":{\"items
\":[{\"value\":\"us-east-1\"}]}},{\"key\":\"aws:ARN\",\"values\":{\"items\":
[{\"value\":\"arn:aws:ec2:us-east-1:111122223333:instance/*\"}]}}]}}}"
}
```


Pour plus d'informations, consultez la section Logique d'évaluation des politiques dans le guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir DecodeAuthorizationMessage dans AWS CLI la référence des commandes.

PowerShell

Outils pour PowerShell

Exemple 1 : Décode les informations supplémentaires contenues dans le contenu du message codé fourni qui a été renvoyé en réponse à une demande. Les informations supplémentaires sont codées car les détails du statut d'autorisation peuvent constituer des informations privilégiées que l'utilisateur qui a demandé l'action ne doit pas voir.

```
Convert-STSAuthorizationMessage -EncodedMessage "...encoded message..."
```

- Pour plus de détails sur l'API, consultez la section DecodeAuthorizationMessage in AWS Tools for PowerShell Cmdlet Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetFederationToken`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetFederationToken`.

CLI

AWS CLI

Pour renvoyer un ensemble d'informations d'identification de sécurité temporaires à l'aide des informations d'identification de la clé d'accès utilisateur IAM

L'`get-federation-token`exemple suivant renvoie un ensemble d'informations d'identification de sécurité temporaires (comprenant un identifiant de clé d'accès, une clé d'accès secrète et un jeton de sécurité) pour un utilisateur. Vous devez appeler l'`GetFederationToken`opération en utilisant les informations d'identification de sécurité à long terme d'un utilisateur IAM.

```
aws sts get-federation-token \
    --name Bob \
    --policy file://myfile.json \
    --policy-arns arn=arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess \
    --duration-seconds 900
```

Contenu de `myfile.json` :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
```

```
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:Describe*",
            "Resource": "*"
        }
    ]
}
```

Sortie :

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": "EXAMPLEpZ2luX2VjEGoaCXVzLXdlc3QtMiJIMEYCIQC/
W9pL5ArQyDD5JwFL3/h5+WGopQ24GEXweNctwhi9sgIhAMkg
+MZE35iWM8s4r5Lr25f9rSTVPFH98G42QQunWMTfKq0DCOP/////////
wEQAxoMNDUyOTI1MTcwNTA3Igxuy3AOpuuoLsk3MJwqgQPg8QOd9HuoClUxq26wnc/nm
+eZLjHDyGf2KUAHK2DuaS/nrGSEXAMPLE",
        "Expiration": "2023-12-20T02:06:07+00:00"
    },
    "FederatedUser": {
        "FederatedUserId": "111122223333:Bob",
        "Arn": "arn:aws:sts::111122223333:federated-user/Bob"
    },
    "PackedPolicySize": 36
}
```

Pour plus d'informations, consultez Demande d'informations d'identification temporaires de sécurité dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir GetFederationToken in AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : demande un jeton fédéré valide pendant une heure en utilisant « Bob » comme nom de l'utilisateur fédéré. Ce nom peut être utilisé pour faire référence au nom d'utilisateur fédéré dans une politique basée sur les ressources (telle qu'une politique de compartiment Amazon S3). La politique IAM fournie, au format JSON, est utilisée pour limiter les autorisations disponibles pour l'utilisateur IAM. La politique fournie ne peut pas accorder plus d'autorisations que celles accordées à l'utilisateur demandeur, les autorisations finales pour l'utilisateur fédéré étant l'ensemble le plus restrictif en fonction de l'intersection entre la politique adoptée et la politique utilisateur IAM.

```
Get-STSFederationToken -Name "Bob" -Policy "...JSON policy..." -DurationInSeconds
 3600
```

- Pour plus de détails sur l'API, voir GetFederationToken in AWS Tools for PowerShell Cmdlet Reference.

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Utilisation **`GetSessionToken`** avec un AWS SDK ou une CLI

Les exemples de code suivants montrent comment utiliser`GetSessionToken`.

Les exemples d'actions sont des extraits de code de programmes de plus grande envergure et doivent être exécutés en contexte. Vous pouvez voir cette action en contexte dans l'exemple de code suivant :

- Obtenir un jeton de session qui nécessite un jeton MFA

CLI

AWS CLI

Pour obtenir un ensemble d'informations d'identification à court terme d'une identité IAM

La commande `get-session-token` suivante extrait un ensemble d'informations d'identification à court terme pour l'identité IAM qui effectue l'appel. Les informations d'identification obtenues peuvent être utilisées pour les requêtes où l'authentification multifactorielle (MFA) est requise par la politique. Les informations d'identification expirent 15 minutes après leur création.

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

Sortie :

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

Pour plus d'informations, consultez Demande d'informations d'identification temporaires de sécurité dans le Guide de l'utilisateur AWS IAM.

- Pour plus de détails sur l'API, voir GetSessionToken in AWS CLI Command Reference.

PowerShell

Outils pour PowerShell

Exemple 1 : renvoie une **`Amazon.RuntimeAWSCredentials`** instance contenant des informations d'identification temporaires valides pendant une période définie. Les informations

d'identification utilisées pour demander des informations d'identification temporaires sont déduites des paramètres par défaut actuels du shell. Pour spécifier d'autres informations d'identification, utilisez les SecretKey paramètres - ProfileName ou - AccessKey /-.

```
Get-STSSessionToken
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemple 2 : renvoie une **Amazon.RuntimeAWSCredentials** instance contenant des informations d'identification temporaires valides pendant une heure. Les informations d'identification utilisées pour effectuer la demande sont obtenues à partir du profil spécifié.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

Exemple 3 : renvoie une **Amazon.RuntimeAWSCredentials** instance contenant des informations d'identification temporaires valides pendant une heure en utilisant le numéro d'identification de l'appareil MFA associé au compte dont les informations d'identification sont spécifiées dans le profil « myprofilename » et la valeur fournie par l'appareil.

```
Get-STSSessionToken -DurationInSeconds 3600 -ProfileName myprofile -SerialNumber
 YourMFADeviceSerialNumber -TokenCode 123456
```

Sortie :

```
AccessKeyId                              Expiration
 SecretAccessKey                         SessionToken
-----------                              ----------
 ---------------                         ------------
EXAMPLEACCESSKEYID                       2/16/2015 9:12:28 PM
 examplesecretaccesskey...               SamPleTokeN.....
```

- Pour plus de détails sur l'API, voir GetSessionToken in AWS Tools for PowerShell Cmdlet Reference.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus sur GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Obtenez un jeton de session en transmettant un jeton MFA et utilisez-le pour répertorier les compartiments Amazon S3 pour le compte.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
```

```
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Pour plus de détails sur l'API, consultez le GetSessionmanuel de référence de l'API Token in AWS SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Scénarios d' AWS STS utilisation des AWS SDK

Les exemples de code suivants vous montrent comment implémenter des scénarios courants AWS STS avec AWS les SDK. Ces scénarios vous montrent comment accomplir des tâches spécifiques en appelant plusieurs fonctions AWS STS. Chaque scénario inclut un lien vers GitHub, où vous pouvez trouver des instructions sur la façon de configurer et d'exécuter le code.

Exemples

- Assumez un rôle IAM qui nécessite un jeton MFA à l' AWS STS aide d'un SDK AWS
- Création d'une URL AWS STS pour les utilisateurs fédérés à l'aide d'un SDK AWS
- Obtenez un jeton de session qui nécessite un jeton MFA à AWS STS l'aide d'un SDK AWS

Assumez un rôle IAM qui nécessite un jeton MFA à l' AWS STS aide d'un SDK AWS

L'exemple de code suivant montre comment assumer un rôle qui nécessite un jeton MFA.

> **Warning**
>
> Afin d'éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

- Créez un rôle IAM qui accorde l'autorisation de répertorier les compartiments Amazon S3.
- Créez un utilisateur IAM autorisé à assumer le rôle uniquement lorsque des informations d'identification MFA sont fournies.
- Enregistrez un périphérique MFA pour l'utilisateur.
- Assumer le rôle et utiliser des informations d'identification temporaires pour répertorier des compartiments S3.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez un utilisateur IAM, enregistrez un dispositif MFA et créez un rôle qui accorde l'autorisation de répertorier des compartiments S3. L'utilisateur n'a que le droit d'assumer le rôle.

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual MFA device.
    Displays the QR code to seed the device.
```

```
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role and requires
MFA.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    For demonstration purposes, the user is created in the same account as the
role,
    but in practice the user would likely be from another account.

    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
resource
                         that has permissions to create users, roles, and
policies
                         in the account.
    :return: The newly created user, user key, virtual MFA device, and role.
    """
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
```

```
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": user.arn},
                        "Action": "sts:AssumeRole",
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
True}},
                    }
                ],
            }
        ),
    )
    print(f"Created role {role.name} that requires MFA.")

    policy = iam_resource.create_policy(
        PolicyName=unique_name("policy"),
        PolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                    }
                ],
```

```
            }
        ),
    )
    role.attach_policy(PolicyArn=policy.arn)
    print(f"Created policy {policy.policy_name} and attached it to the role.")

    user.create_policy(
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "sts:AssumeRole",
                        "Resource": role.arn,
                    }
                ],
            }
        ),
    )
    print(
        f"Created an inline policy for {user.name} that lets the user assume "
        f"the role."
    )

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device, role
```

Montrez qu'assumer le rôle sans jeton MFA n'est pas autorisé.

```
def try_to_assume_role_without_mfa(assume_role_arn, session_name, sts_client):
    """
    Shows that attempting to assume the role without sending MFA credentials
 results
    in an AccessDenied error.
```

```
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role to assume.
    :param session_name: The name of the STS session.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    print(f"Trying to assume the role without sending MFA credentials...")
    try:
        sts_client.assume_role(RoleArn=assume_role_arn,
 RoleSessionName=session_name)
        raise RuntimeError("Expected AccessDenied error.")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Got AccessDenied.")
        else:
            raise
```

Assumez le rôle qui accorde l'autorisation de répertorier des compartiments S3, en transmettant le jeton MFA requis, et montrez que les compartiments peuvent être répertoriés.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
```

```
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

Détruisez les ressources créées pour la démo.

```
def teardown(user, virtual_mfa_device, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        policy_name = attached.policy_name
        role.detach_policy(PolicyArn=attached.arn)
        attached.delete()
        print(f"Detached and deleted {policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
```

```
        key.delete()
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

Exécutez ce scénario à l'aide des fonctions définies précédemment.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device, role = setup(iam_resource)
    print(f"Created {user.name} and {role.name}.")
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
        )
        try_to_assume_role_without_mfa(role.arn, "demo-sts-session", sts_client)
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_from_assumed_role_with_mfa(
            role.arn,
            "demo-sts-session",
            virtual_mfa_device.serial_number,
            mfa_totp,
            sts_client,
        )
    finally:
        teardown(user, virtual_mfa_device, role)
        print("Thanks for watching!")
```

- Pour plus de détails sur l'API, consultez AssumeRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Création d'une URL AWS STS pour les utilisateurs fédérés à l'aide d'un SDK AWS

L'exemple de code suivant illustre comment :

- Créez un rôle IAM qui accorde un accès en lecture seule aux ressources Amazon S3 du compte actuel.
- Obtenez un jeton de sécurité auprès du point de terminaison AWS de la fédération.
- Créez une URL qui puisse être utilisée pour accéder à la console avec des informations d'identification fédérées.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez un rôle qui accorde un accès en lecture seule aux ressources Amazon S3 du compte actuel.

```
def setup(iam_resource):
    """
    Creates a role that can be assumed by the current user.
    Attaches a policy that allows only Amazon S3 read-only access.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 instance
```

```
                        that has the permission to create a role.
    :return: The newly created role.
    """
    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": iam_resource.CurrentUser().arn},
                        "Action": "sts:AssumeRole",
                    }
                ],
            }
        ),
    )
    role.attach_policy(PolicyArn="arn:aws:iam::aws:policy/
AmazonS3ReadOnlyAccess")
    print(f"Created role {role.name}.")

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return role
```

Obtenez un jeton de sécurité auprès du point de terminaison de la AWS fédération et créez une URL qui peut être utilisée pour accéder à la console avec des informations d'identification fédérées.

```
def construct_federated_url(assume_role_arn, session_name, issuer, sts_client):
    """
    Constructs a URL that gives federated users direct access to the AWS
 Management
    Console.

    1. Acquires temporary credentials from AWS Security Token Service (AWS STS)
 that
```

```
       can be used to assume a role with limited permissions.
   2. Uses the temporary credentials to request a sign-in token from the
      AWS federation endpoint.
   3. Builds a URL that can be used in a browser to navigate to the AWS
federation
      endpoint, includes the sign-in token for authentication, and redirects to
      the AWS Management Console with permissions defined by the role that was
      specified in step 1.

   :param assume_role_arn: The role that specifies the permissions that are
granted.
                           The current user must have permission to assume the
role.
   :param session_name: The name for the STS session.
   :param issuer: The organization that issues the URL.
   :param sts_client: A Boto3 STS instance that can assume the role.
   :return: The federated URL.
   """
   response = sts_client.assume_role(
       RoleArn=assume_role_arn, RoleSessionName=session_name
   )
   temp_credentials = response["Credentials"]
   print(f"Assumed role {assume_role_arn} and got temporary credentials.")

   session_data = {
       "sessionId": temp_credentials["AccessKeyId"],
       "sessionKey": temp_credentials["SecretAccessKey"],
       "sessionToken": temp_credentials["SessionToken"],
   }
   aws_federated_signin_endpoint = "https://signin.aws.amazon.com/federation"

   # Make a request to the AWS federation endpoint to get a sign-in token.
   # The requests.get function URL-encodes the parameters and builds the query
string
   # before making the request.
   response = requests.get(
       aws_federated_signin_endpoint,
       params={
           "Action": "getSigninToken",
           "SessionDuration": str(datetime.timedelta(hours=12).seconds),
           "Session": json.dumps(session_data),
       },
   )
   signin_token = json.loads(response.text)
```

```
    print(f"Got a sign-in token from the AWS sign-in federation endpoint.")

    # Make a federated URL that can be used to sign into the AWS Management
 Console.
    query_string = urllib.parse.urlencode(
        {
            "Action": "login",
            "Issuer": issuer,
            "Destination": "https://console.aws.amazon.com/",
            "SigninToken": signin_token["SigninToken"],
        }
    )
    federated_url = f"{aws_federated_signin_endpoint}?{query_string}"
    return federated_url
```

Détruisez les ressources créées pour la démo.

```
def teardown(role):
    """
    Removes all resources created during setup.

    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        role.detach_policy(PolicyArn=attached.arn)
        print(f"Detached {attached.policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
```

Exécutez ce scénario à l'aide des fonctions définies précédemment.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the AWS Security Token Service federated URL demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
```

```
    role = setup(iam_resource)
    sts_client = boto3.client("sts")
    try:
        federated_url = construct_federated_url(
            role.arn, "AssumeRoleDemoSession", "example.org", sts_client
        )
        print(
            "Constructed a federated URL that can be used to connect to the "
            "AWS Management Console with role-defined permissions:"
        )
        print("-" * 88)
        print(federated_url)
        print("-" * 88)
        _ = input(
            "Copy and paste the above URL into a browser to open the AWS "
            "Management Console with limited permissions. When done, press "
            "Enter to clean up and complete this demo."
        )
    finally:
        teardown(role)
        print("Thanks for watching!")
```

- Pour plus de détails sur l'API, consultez AssumeRolele AWS manuel de référence de l'API SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Obtenez un jeton de session qui nécessite un jeton MFA à AWS STS l'aide d'un SDK AWS

L'exemple de code suivant montre comment obtenir un jeton de session nécessitant un jeton MFA.

> **Warning**
>
> Afin d'éviter les risques de sécurité, n'employez pas les utilisateurs IAM pour l'authentification lorsque vous développez des logiciels spécialisés ou lorsque vous travaillez avec des

données réelles. Préférez la fédération avec un fournisseur d'identité tel que AWS IAM Identity Center.

- Créez un rôle IAM qui accorde l'autorisation de répertorier les compartiments Amazon S3.
- Créez un utilisateur IAM autorisé à assumer le rôle uniquement lorsque des informations d'identification MFA sont fournies.
- Enregistrez un périphérique MFA pour l'utilisateur.
- Fournissez des informations d'identification MFA pour obtenir un jeton de session et utilisez des informations d'identification temporaires pour répertorier les compartiments S3.

Python

SDK pour Python (Boto3)

> **Note**
>
> Il y en a plus à ce sujet GitHub. Trouvez l'exemple complet et découvrez comment le configurer et l'exécuter dans le référentiel d'exemples de code AWS.

Créez un utilisateur IAM, enregistrez un dispositif MFA et créez un rôle qui accorde l'autorisation de laisser l'utilisateur répertorier des compartiments S3 uniquement lorsque les informations d'identification MFA sont utilisées.

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual multi-factor authentication (MFA) device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates an inline policy for the user that lets the user list Amazon S3
 buckets,
    but only when MFA credentials are used.

    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
```

```
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
resource
                         that has permissions to create users, MFA devices, and
                         policies in the account.
    :return: The newly created user, user key, and virtual MFA device.
    """
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    user.create_policy(
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
```

```
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
 True}},
                    }
                ],
            }
        ),
    )
    print(
        f"Created an inline policy for {user.name} that lets the user list
 buckets, "
        f"but only when MFA credentials are present."
    )

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device
```

Obtenez des informations d'identification de session temporaires en transmettant un jeton MFA et utilisez les informations d'identification pour répertorier les compartiments S3 pour le compte.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
```

```
                                device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

Détruisez les ressources créées pour la démo.

```
def teardown(user, virtual_mfa_device):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo MFA device.
    """
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
```

```
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

Exécutez ce scénario à l'aide des fonctions définies précédemment.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device = setup(iam_resource)
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
        )
        try:
            print("Listing buckets without specifying MFA credentials.")
            list_buckets_with_session_token_with_mfa(None, None, sts_client)
        except ClientError as error:
            if error.response["Error"]["Code"] == "AccessDenied":
                print("Got expected AccessDenied error.")
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_with_session_token_with_mfa(
            virtual_mfa_device.serial_number, mfa_totp, sts_client
        )
    finally:
        teardown(user, virtual_mfa_device)
        print("Thanks for watching!")
```

- Pour plus de détails sur l'API, consultez le GetSessionmanuel de référence de l'API Token in AWS SDK for Python (Boto3).

Pour obtenir la liste complète des guides de développement du AWS SDK et des exemples de code, consultezUtilisation d'IAM avec un SDK AWS. Cette rubrique comprend également des informations sur le démarrage et sur les versions précédentes de SDK.

Sécurité dans IAM et AWS STS

La sécurité du cloud AWS est la priorité absolue. En tant que AWS client, vous bénéficiez de centres de données et d'architectures réseau conçus pour répondre aux exigences des entreprises les plus sensibles en matière de sécurité.

La sécurité est une responsabilité partagée entre vous AWS et vous. Le modèle de responsabilité partagée décrit cela comme la sécurité du cloud et la sécurité dans le cloud :

- Sécurité du cloud : AWS est chargée de protéger l'infrastructure qui exécute les AWS services dans le AWS cloud. AWS vous fournit également des services que vous pouvez utiliser en toute sécurité. Des auditeurs tiers testent et vérifient régulièrement l'efficacité de notre sécurité dans le cadre des programmes de AWS conformité Programmes de de conformité. Pour en savoir plus sur les programmes de conformité qui s'appliquent à AWS Identity and Access Management (IAM), voir AWS Services concernés par programme de conformitéAWS .
- Sécurité dans le cloud — Votre responsabilité est déterminée par le AWS service que vous utilisez. Vous êtes également responsable d'autres facteurs, y compris la sensibilité de vos données, les exigences de votre entreprise, ainsi que la législation et la réglementation applicables.

Cette documentation vous aide à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation de AWS Identity and Access Management (IAM) et AWS Security Token Service (AWS STS). Les rubriques suivantes vous montrent comment configurer l'IAM et comment AWS STS atteindre vos objectifs de sécurité et de conformité. Vous apprendrez également à utiliser d'autres AWS services qui vous aident à surveiller et à sécuriser vos ressources IAM.

Table des matières

- AWS informations d'identification de sécurité
- AWS directives relatives aux audits de sécurité
- Protection des données dans AWS Identity and Access Management
- Connexion et surveillance AWS Identity and Access Management
- Validation de la conformité pour AWS Identity and Access Management
- Résilience dans AWS Identity and Access Management
- Sécurité de l’infrastructure dans AWS Identity and Access Management
- Analyse de configuration et de vulnérabilité dans AWS Identity and Access Management

- AWS politiques gérées pour AWS Identity and Access Management Access Analyzer

AWS informations d'identification de sécurité

Lorsque vous interagissez avec AWS, vous spécifiez vos informations de AWS sécurité pour vérifier qui vous êtes et si vous êtes autorisé à accéder aux ressources que vous demandez. AWS utilise les informations d'identification de sécurité pour authentifier et autoriser vos demandes.

Par exemple, si vous souhaitez télécharger un fichier protégé à partir d'un compartiment Amazon Simple Storage Service (Amazon S3), vos informations d'identification doivent autoriser cet accès. Si vos informations d'identification n'indiquent pas que vous êtes autorisé à télécharger le fichier, AWS refuse votre demande. Cependant, vos informations d'identification de AWS sécurité ne sont pas nécessaires pour télécharger un fichier dans un compartiment Amazon S3 partagé publiquement.

Il existe différents types d'utilisateurs dans AWS. Tous les AWS utilisateurs possèdent des identifiants de sécurité. Il y a le propriétaire du compte (utilisateur root) utilisateurs dans IAM Identity Center, les utilisateurs fédérés et les utilisateurs IAM.

Les utilisateurs disposent d'informations d'identification de sécurité à long terme ou temporaires. L'utilisateur root, l'utilisateur IAM et les clés d'accès possèdent des informations d'identification de sécurité à long terme qui n'expirent pas. Pour protéger les informations d'identification à long terme, il convient de mettre en place des processus pour gérer les clés d'accès, modifier les mots de passe et activer MFA.

Les rôles IAM et les utilisateurs dans IAM Identity Center utilisateurs fédérés disposent d'informations d'identification de sécurité temporaires. Les informations d'identification de sécurité temporaires expirent après une période définie ou lorsque l'utilisateur met fin à sa session. Les informations d'identification temporaires fonctionnent de manière presque identique aux informations d'identification des clés d'accès à long terme, à quelques différences près :

- Les informations d'identification de sécurité temporaires sont à court terme, comme leur nom l'indique. Elles peuvent être configurées pour être valides de quelques minutes à plusieurs heures. Une fois les informations d'identification AWS expirées, il ne les reconnaît plus ou n'autorise aucun type d'accès à partir des demandes d'API effectuées avec elles.
- Les informations d'identification de sécurité temporaires ne sont pas stockées avec l'utilisateur, mais sont générées automatiquement et fournies à l'utilisateur sur demande. Lorsque (ou même avant) les informations d'identification de sécurité temporaires arrivent à expiration, l'utilisateur peut en demander de nouvelles, tant qu'il y est autorisé.

Par conséquent, les informations d'identification temporaires présentent les avantages suivants par rapport aux informations d'identification à long terme :

- Il n'est pas nécessaire de distribuer ou d'intégrer des informations d'identification AWS de sécurité à long terme dans une application.
- Vous pouvez donner accès à vos AWS ressources aux utilisateurs sans avoir à définir leur AWS identité. Les informations d'identification temporaires servent de base aux rôles et à la fédération d'identité.
- Les informations d'identification de sécurité temporaires ont une durée de vie limitée. Il n'est donc pas nécessaire de les mettre à jour ou de les révoquer explicitement lorsqu'elles deviennent obsolètes. Une fois que les informations d'identification de sécurité temporaires arrivent à expiration, elles ne peuvent pas être réutilisées. Vous pouvez spécifier le délai de validité des informations d'identification, jusqu'à une certaine limite.

Considérations sur la sécurité

Nous vous recommandons de tenir compte des informations suivantes lorsque vous déterminez les mesures de sécurité pour votre Compte AWS :

- Lorsque vous créez un Compte AWS, nous créons le compte utilisateur root. Les informations d'identification de l'utilisateur root (propriétaire du compte) permettent un accès complet à toutes les ressources du compte. La première tâche que vous effectuez avec l'utilisateur root consiste à accorder à un autre utilisateur des autorisations administratives Compte AWS afin de minimiser l'utilisation de l'utilisateur root.
- Vous ne pouvez pas utiliser des politiques IAM pour refuser l'accès aux ressources de manière explicite à l'utilisateur root. Vous ne pouvez utiliser une politique AWS Organizations de contrôle des services (SCP) que pour limiter les autorisations de l'utilisateur root.
- Si vous oubliez ou perdez votre mot de passe root, vous devez avoir accès à l'adresse e-mail associée à votre compte pour le réinitialiser.
- Si vous perdez vos clés d'accès de l'utilisateur root, vous devez vous connecter à votre compte en tant qu'utilisateur root pour en créer de nouvelles.
- N'utilisez pas l'utilisateur root de vos tâches courantes. Optez pour effectuer les tâches que seul l'utilisateur root peut effectuer. Pour obtenir la liste complète des tâches qui nécessitent que vous vous connectiez en tant qu'utilisateur root, veuillez consulter Tâches nécessitant les informations d'identification de l'utilisateur root.

- Les informations d'identification de sécurité sont spécifiques au compte. Si vous avez accès à plusieurs Comptes AWS, vous disposez d'informations d'identification distinctes pour chaque compte.
- Les politiques déterminent les actions qu'un utilisateur, un rôle ou un membre d'un groupe d'utilisateurs peut effectuer, sur quelles AWS ressources et dans quelles conditions. À l'aide de politiques, vous pouvez contrôler en toute sécurité l'accès Services AWS et les ressources de votre Compte AWS. Si vous devez modifier ou révoquer des autorisations en réponse à un événement de sécurité, vous supprimez ou modifiez les politiques au lieu de modifier directement l'identité.
- Veillez à enregistrer les informations de connexion de votre utilisateur IAM Emergency Access et toutes les clés d'accès que vous avez créées pour un accès programmatique dans un emplacement sécurisé. Si vous perdez vos clés d'accès, vous devez vous connecter à votre compte pour en créer de nouvelles.
- Nous vous recommandons vivement d'utiliser les informations d'identification temporaires fournies par les rôles IAM et les utilisateurs fédérés plutôt que les informations d'identification à long terme fournies par les utilisateurs IAM et les clés d'accès.

Identité fédérée

Les identités fédérées sont des utilisateurs dotés d'identités externes auxquels sont accordées des AWS informations d'identification temporaires qu'ils peuvent utiliser pour accéder à Compte AWS des ressources sécurisées. Les identités externes peuvent provenir d'un magasin d'identités d'entreprise (tel que LDAP ou Windows Active Directory) ou d'une partie tierce (Login with Amazon, Facebook ou Google, par exemple). Les identités fédérées ne se connectent pas au portail AWS Management Console ou n' AWS accèdent pas.

Pour permettre aux identités fédérées de se connecter à AWS, vous devez créer une URL personnalisée qui inclut `https://signin.aws.amazon.com/federation` Pour plus d'informations, consultez Activation de l'accès à la AWS console par un courtier d'identité personnalisé.

Pour plus d'informations sur les identités fédérées, consultez Fournisseurs d'identité et fédération.

Authentification multifactorielle (MFA)

L'authentification multi-facteurs (Multi-Factor Authentication, MFA) fournit un niveau de sécurité supplémentaire pour les utilisateurs pouvant accéder à votre Compte AWS. Pour une sécurité optimale, nous vous recommandons d'activer MFA pour les informations d'identification de l'

Utilisateur racine d'un compte AWS et pour tous les utilisateurs IAM. Pour plus d'informations, consultez Utilisation de l'authentification multifactorielle (MFA) dans AWS.

Lorsque vous activez l'authentification multifacteur et que vous vous connectez à votre Compte AWS compte, vous êtes invité à saisir vos informations d'identification, ainsi qu'une réponse générée par un appareil MFA, telle qu'un code, un toucher ou un scan biométrique. Lorsque vous ajoutez la MFA, vos Compte AWS paramètres et vos ressources sont plus sécurisés.

Par défaut, l'authentification MFA n'est pas activée. Vous pouvez activer et gérer les périphériques MFA pour l' Utilisateur racine d'un compte AWS en accédant à la page Informations d'identification de sécurité ou au tableau de bord IAM dans l' AWS Management Console. Pour en savoir plus sur l'activation de l'authentification MFA pour les utilisateurs IAM, consultez Activation des appareils MFA pour les utilisateurs de AWS.

Pour plus d'informations sur la connexion aux dispositifs d'authentification multifactorielle (MFA), consultez Utilisation de dispositifs MFA avec votre page de connexion IAM.

Accès par programmation

Vous fournissez vos clés AWS d'accès pour effectuer des appels programmatiques vers AWS ou pour utiliser le AWS Command Line Interface ou AWS Tools for PowerShell. Nous vous recommandons d'utiliser des clés d'accès à court terme si possible.

Lorsque vous créez une clé d'accès à long terme, vous générez l'ID de clé d'accès (par exemple, `AKIAIOSFODNN7EXAMPLE`) et la clé d'accès secrète (par exemple, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`) sous la forme d'un ensemble. La clé d'accès secrète est accessible en téléchargement uniquement au moment de sa création. Si vous ne téléchargez pas votre clé d'accès secrète ou si vous la perdez, vous devrez la recréer.

Dans de nombreux cas, vous n'avez pas besoin de clés d'accès à long terme qui n'expirent jamais (comme lorsque vous créez des clés d'accès pour un utilisateur IAM). Au lieu de cela, vous pouvez créer des rôles IAM et générer des informations d'identification de sécurité temporaires. Les informations d'identification de sécurité temporaires comprennent non seulement un ID de clé d'accès et une clé d'accès secrète, mais également un jeton de sécurité qui indique la date d'expiration des informations d'identification. Après leur expiration, elles ne sont plus valides.

Les identifiants de clé d'accès commençant par `AKIA` sont des clés d'accès à long terme pour un utilisateur IAM ou un utilisateur Compte AWS root. Les identifiants de clé d'accès commençant par `ASIA` sont des clés d'accès aux informations d'identification temporaires que vous créez à l'aide d' AWS STS opérations.

Les utilisateurs ont besoin d'un accès programmatique s'ils souhaitent interagir avec AWS l'extérieur du AWS Management Console. La manière d'accorder un accès programmatique dépend du type d'utilisateur qui y accède AWS.

Pour accorder aux utilisateurs un accès programmatique, choisissez l'une des options suivantes.

Quel utilisateur a besoin d'un accès programmatique ?	Pour	Par
Identité de la main-d'œuvre (Utilisateurs gérés dans IAM Identity Center)	Utilisez des informations d'identification temporaires pour signer les demandes programmatiques adressées aux AWS CLI AWS SDK ou AWS aux API.	Suivez les instructions de l'interface que vous souhaitez utiliser. • Pour le AWS CLI, voir Configuration du AWS CLI à utiliser AWS IAM Identity Center dans le guide de AWS Command Line Interface l'utilisateur. • Pour les AWS SDK, les outils et les AWS API, consultez la section Authentification IAM Identity Center dans le Guide de référence AWS des SDK et des outils.
IAM	Utilisez des informations d'identification temporaires pour signer les demandes programmatiques adressées aux AWS CLI AWS SDK ou AWS aux API.	Suivez les instructions de la section Utilisation d'informations d'identification temporaires avec AWS les ressources du Guide de l'utilisateur IAM.
IAM	(Non recommandé) Utilisez des informations d'identification à long terme pour signer les AWS CLI	Suivez les instructions de l'interface que vous souhaitez utiliser.

Quel utilisateur a besoin d'un accès programmatique ?	Pour	Par
	demandes programmatiques adressées aux AWS SDK ou AWS aux API.	• Pour le AWS CLI, voir Authentification à l'aide des informations d'identification utilisateur IAM dans le Guide de l'AWS Command Line Interface utilisateur. • Pour les AWS SDK et les outils, voir Authentifier à l'aide d'informations d'identification à long terme dans le Guide de AWS référence des SDK et des outils. • Pour les AWS API, consultez la section Gestion des clés d'accès pour les utilisateurs IAM dans le guide de l'utilisateur IAM.

Alternatives aux clés d'accès à long terme

Pour de nombreux cas d'utilisation courants, il existe des alternatives aux clés d'accès à long terme. Pour améliorer la sécurité de votre compte, tenez compte des points suivants.

- N'intégrez pas de clés d'accès à long terme ni de clés d'accès secrètes dans le code de votre application ou dans un référentiel de code. Utilisez AWS Secrets Manager plutôt une solution de gestion des secrets ou une autre solution de gestion des secrets, afin de ne pas avoir à coder les clés en dur en texte brut. L'application ou le client peut ensuite récupérer des secrets en cas de besoin. Pour plus d'informations, voir Qu'est-ce que c'est AWS Secrets Manager ? dans le guide de AWS Secrets Manager l'utilisateur.

- Utilisez les rôles IAM pour générer des informations d'identification de sécurité temporaires dans la mesure du possible : utilisez toujours des mécanismes permettant de délivrer des identifiants de sécurité temporaires, lorsque cela est possible, plutôt que des clés d'accès à long terme. Les

informations d'identification de sécurité temporaires sont plus sécurisées car elles ne sont pas stockées avec l'utilisateur, mais sont générées automatiquement et fournies à l'utilisateur sur demande. Les informations d'identification de sécurité temporaires ont une durée de vie limitée, ce qui vous évite d'avoir à les gérer ou à les mettre à jour. Les mécanismes qui fournissent des clés d'accès temporaires incluent les rôles IAM ou l'authentification d'un utilisateur d'IAM Identity Center. Pour les machines qui s'exécutent à l'extérieur, AWS vous pouvez utiliser AWS Identity and Access Management Roles Anywhere.

- Utilisez des alternatives aux clés d'accès à long terme pour l' AWS Command Line Interface (AWS CLI) ou **`aws-shell`** — Les alternatives incluent ce qui suit.
  - AWS CloudShellest un shell pré-authentifié basé sur un navigateur que vous pouvez lancer directement depuis le. AWS Management Console Vous pouvez exécuter AWS CLI des commandes par Services AWS le biais de votre shell préféré (shell Bash, Powershell ou Z). Dans ce cas, il n'est pas nécessaire de télécharger ou d'installer des outils de ligne de commande. Pour plus d'informations, consultez Présentation d' AWS CloudShell dans le Guide de l'utilisateur AWS CloudShell .
  - AWS CLI Intégration de la version 2 avec AWS IAM Identity Center (IAM Identity Center). Vous pouvez authentifier les utilisateurs et fournir des informations d'identification à court terme pour exécuter des AWS CLI commandes. Pour en savoir plus, consultez les sections Intégration AWS CLI à IAM Identity Center dans le guide de AWS IAM Identity Center l'utilisateur et Configuration du centre AWS CLI pour utiliser IAM Identity Center dans le guide de l'AWS Command Line Interface utilisateur.
- Ne créez pas de clés d'accès à long terme pour les utilisateurs humains qui ont besoin d'accéder aux applications. Services AWS Sinon, IAM Identity Center peut générer des informations d'accès temporaires auxquelles les utilisateurs externes de votre IdP peuvent accéder. Services AWS Il n'est donc plus nécessaire de créer et de gérer des informations d'identification à long terme dans IAM. Dans IAM Identity Center, créez un ensemble d'autorisations IAM Identity Center qui accorde l'accès aux utilisateurs IdP externes. Attribuez ensuite un groupe d'IAM Identity Center aux autorisations définies dans le champ sélectionné Comptes AWS. Pour plus d'informations, consultez les sections Qu'est-ce que c'est AWS IAM Identity Center, Se connecter à votre fournisseur d'identité externe et Ensembles d'autorisations dans le guide de AWS IAM Identity Center l'utilisateur.
- Ne stockez pas de clés d'accès à long terme dans un service AWS informatique. Attribuez plutôt un rôle IAM aux ressources informatiques. Cela fournit automatiquement des informations d'identification temporaires pour accorder l'accès. Par exemple, lorsque vous créez un profil d'instance attaché à une instance Amazon EC2, vous pouvez attribuer un AWS rôle à l'instance

et le mettre à la disposition de toutes ses applications. Un profil d'instance contient le rôle et permet aux programmes qui s'exécutent sur l'instance Amazon EC2 d'obtenir des informations d'identification temporaires. Pour en savoir plus, consultez Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2.

Accès à AWS l'aide de vos AWS informations d'identification

AWS nécessite différents types d'identifiants de sécurité, en fonction de votre mode d'accès AWS et du type d' AWS utilisateur que vous êtes. Par exemple, vous utilisez les informations de connexion pour AWS Management Console pendant que vous utilisez les touches d'accès pour passer des appels par programmation à AWS. De plus, chaque identité que vous utilisez, qu'il s'agisse de l'utilisateur racine du compte, d'un utilisateur AWS Identity and Access Management (IAM), d'un AWS IAM Identity Center utilisateur ou d'une identité fédérée, possède des informations d'identification uniques. AWS

Pour step-by-step obtenir des instructions sur la procédure de connexion en AWS fonction de votre type d'utilisateur, consultez Comment vous connecter AWS dans le guide de l'utilisateur de AWS connexion.

AWS directives relatives aux audits de sécurité

Effectuez régulièrement un audit de votre configuration de sécurité pour vérifier qu'elle répond aux besoins actuels de votre entreprise. Un audit vous donne la possibilité de supprimer les utilisateurs, les rôles, les groupes et les politiques IAM inutiles, et de vérifier que vos utilisateurs et vos logiciels ne disposent pas d'autorisations excessives.

Vous trouverez ci-dessous des directives pour examiner et surveiller systématiquement vos AWS ressources afin de respecter les meilleures pratiques en matière de sécurité.

> **Tip**
>
> Vous pouvez surveillez votre utilisation d'IAM, conformément aux bonnes pratiques de sécurité, à l'aide d'AWS Security Hub. Security Hub utilise des contrôles de sécurité pour évaluer les configurations des ressources et les normes de sécurité afin de vous aider à respecter divers cadres de conformité. Pour plus d'informations sur l'utilisation de Security Hub pour évaluer les ressources IAM, consultez Contrôles d'AWS Identity and Access Management dans le Guide de l'utilisateur AWS Security Hub .

Table des matières

- Quand effectuer un audit de sécurité ?
- Consignes pour l'audit
- Vérifiez les informations d'identification AWS de votre compte
- Examen de vos utilisateurs IAM
- Examen de vos groupes IAM
- Examen de vos rôles IAM
- Examen de vos fournisseurs IAM pour SAML et OpenID Connect (OIDC)
- Examen de vos applications mobiles
- Conseils pour l'examen des politiques IAM

Quand effectuer un audit de sécurité ?

Effectuez un audit de votre configuration de sécurité dans les cas suivants :

- De manière régulière. Exécutez les étapes décrites dans ce document à intervalles réguliers, à titre de bonne pratique de sécurité.
- Si des changements se produisent dans votre organisation, par exemple lorsque des personnes quittent l'organisation.
- Si vous avez cessé d'utiliser un ou plusieurs AWS services individuels pour vérifier que vous avez supprimé les autorisations dont les utilisateurs de votre compte n'ont plus besoin.
- Si vous avez ajouté ou supprimé des logiciels dans vos comptes, tels que des applications sur des instances Amazon EC2, des AWS OpsWorks piles, des AWS CloudFormation modèles, etc.
- Si vous pensez qu'une personne non autorisée a pu avoir accès à votre compte.

Consignes pour l'audit

Lorsque vous examinez la configuration de sécurité de votre compte, suivez ces consignes :

- Soyez minutieux. Regardez tous les aspects de votre configuration de sécurité, y compris ceux qui sont rarement utilisés.
- Ne faites pas de suppositions. Si vous ne connaissez pas vraiment certains aspects de votre configuration de sécurité (par exemple, le raisonnement qui justifie une stratégie particulière ou

l'existence d'un rôle), étudiez les besoins de l'entreprise jusqu'à ce que vous compreniez le risque potentiel.

- Simplifiez les choses. Pour faciliter l'audit (et la gestion), utilisez les groupes et les rôles IAM, les schémas d'affectation de nom cohérents et les stratégies simples.

Vérifiez les informations d'identification AWS de votre compte

Procédez comme suit lorsque vous auditez les informations d'identification de votre AWS compte :

1. Si vous avez des clés d'accès pour votre utilisateur root que vous n'utilisez pas, vous pouvez les supprimer. Nous vous recommandons vivement de ne pas utiliser les clés d'accès root dans le cadre de votre travail quotidien AWS, mais plutôt d'utiliser des utilisateurs dotés d'informations d'identification temporaires, telles que utilisateurs dans IAM Identity Center.
2. Si vous avez besoin de clés d'accès de votre compte, assurez-vous de les mettre à jour en cas de besoin.

Examen de vos utilisateurs IAM

Exécutez les étapes suivantes lorsque vous effectuez un audit de vos utilisateurs IAM existants :

1. Répertoriez vos utilisateurs, puis supprimez les utilisateurs qui ne sont pas nécessaires.
2. Supprimez les utilisateurs des groupes auxquels ils n'ont pas besoin d'accéder.
3. Examinez les stratégies associées aux groupes dans lesquels se trouve l'utilisateur. veuillez consulter Conseils pour l'examen des politiques IAM.
4. Supprimez les informations d'identification de sécurité inutiles pour l'utilisateur ou susceptibles d'avoir été exposées. Par exemple, un utilisateur IAM utilisé pour une application n'a pas besoin de mot de passe (nécessaire uniquement pour se connecter à des AWS sites Web). De même, si un utilisateur n'utilise pas de clés d'accès, il n'y a aucune raison qu'il en ait une. Pour plus d'informations, consultez Gestion des mots de passe des utilisateurs IAM et Gestion des clés d'accès pour les utilisateurs IAM.

   Vous pouvez générer et télécharger un rapport sur les informations d'identification qui répertorie tous les utilisateurs IAM de votre compte et le statut de leurs diverses informations d'identification, notamment leurs mots de passe, clés d'accès et dispositifs MFA. Pour les mots de passe et les clés d'accès, le rapport d'informations d'identification indique la date et l'heure auxquelles le mot de passe ou la clé d'accès a été utilisé(e) pour la dernière fois. Envisagez de supprimer de votre

compte les informations d'identification qui n'ont pas été utilisées récemment. (Ne supprimez pas votre utilisateur d'accès d'urgence.) Pour plus d'informations, consultez la section Obtenir des rapports d'identification pour votre AWS compte.

5. Mettez à jour les mots de passe et les clés d'accès lorsque cela est nécessaire pour les cas d'utilisation nécessitant des informations d'identification à long terme. Pour plus d'informations, consultez Gestion des mots de passe des utilisateurs IAM et Gestion des clés d'accès pour les utilisateurs IAM.
6. Il est recommandé d'obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder à AWS l'aide d'informations d'identification temporaires. Si possible, passez du statut d'utilisateurs IAM à celui d'utilisateurs fédérés, tels que les utilisateurs d'IAM Identity Center. Conservez le nombre minimum d'utilisateurs IAM requis pour vos applications.

Examen de vos groupes IAM

Exécutez les étapes suivantes lorsque vous effectuez un audit de vos groupes IAM :

1. Répertoriez vos groupes, puis supprimez les groupes que vous n'utilisez pas.
2. Examinez les utilisateurs de chaque groupe et supprimez les utilisateurs qui n’y appartiennent pas.
3. Examinez les stratégies associées au groupe. veuillez consulter Conseils pour l'examen des politiques IAM.

Examen de vos rôles IAM

Exécutez les étapes suivantes lorsque vous effectuez un audit de vos rôles IAM :

1. Répertoriez vos rôles, puis supprimez les rôles que vous n'utilisez pas.
2. Examinez la stratégie d'approbation du rôle. Assurez-vous de connaître le principal et de comprendre pourquoi ce compte ou cet utilisateur doit être capable d'assumer le rôle.
3. Examinez la stratégie d'accès du rôle pour vous assurer qu'elle accorde les autorisations appropriées à la personne qui assume le rôle (voir Conseils pour l'examen des politiques IAM).

Examen de vos fournisseurs IAM pour SAML et OpenID Connect (OIDC)

Si vous avez créé une entité IAM pour établir une relation de confiance avec un fournisseur d'identité (IdP) SAML ou OIDC, procédez comme suit :

1. Supprimez les fournisseurs inutilisés.
2. Téléchargez et consultez les documents de AWS métadonnées pour chaque IdP SAML et assurez-vous qu'ils reflètent les besoins actuels de votre entreprise.
3. Obtenez les derniers documents de métadonnées auprès du SAML IdPs et mettez à jour le fournisseur dans IAM.

Examen de vos applications mobiles

Si vous avez créé une application mobile qui envoie des demandes à AWS, procédez comme suit :

1. Vérifiez que l'application mobile ne contient pas de clés d'accès incorporées, même si elles se trouvent dans le stockage chiffré.
2. Obtenez des informations d'identification temporaires pour l'application à l'aide des API conçues à cette fin.

> **Note**
>
> Nous vous recommandons d'utiliser Amazon Cognito pour gérer l'identité de l'utilisateur dans votre application. Ce service vous permet d'authentifier les utilisateurs à l'aide de Login with Amazon, Facebook, Google ou tout autre fournisseur d'identité compatible avec OpenID Connect (OIDC). Pour plus d'informations sur les pools d'identité Amazon Cognito, consultez le Manuel du développeur Amazon Cognito

Conseils pour l'examen des politiques IAM

Les politiques sont puissantes et subtiles, il est donc important d'étudier et de comprendre les autorisations octroyées par chaque politique. Utilisez les consignes suivantes lors de l'examen des stratégies :

- Attachez des politiques à des groupes ou à des rôles plutôt qu'à des utilisateurs individuels. Si un utilisateur individuel dispose d'une stratégie, assurez-vous de comprendre pourquoi cet utilisateur a besoin d'une stratégie.
- Assurez-vous que les utilisateurs, les groupes d'utilisateurs et les rôles IAM disposent des autorisations dont ils ont besoin et n'ont aucune autorisation supplémentaire.

- Utilisez le simulateur de politiques IAM pour tester les politiques associées à des utilisateurs ou des groupes.
- N'oubliez pas que les autorisations d'un utilisateur sont le résultat de toutes les politiques applicables, à la fois des politiques basées sur l'identité (sur les utilisateurs, les groupes ou les rôles) et des politiques basées sur les ressources (sur des ressources telles que les compartiments Amazon S3, les files d'attente Amazon SQS, les rubriques Amazon SNS et les clés). AWS KMS Il est important d'examiner toutes les stratégies qui s'appliquent à un utilisateur et de comprendre l'ensemble complet des autorisations octroyées à un utilisateur individuel.
- Sachez qu'en autorisant un utilisateur à créer un utilisateur, un groupe, un rôle ou une politique IAM et à associer une politique à l'entité principale, toutes les autorisations à toutes les ressources de votre compte lui sont en fait octroyées. Les utilisateurs qui peuvent créer des politiques et les associer à un utilisateur, un groupe ou un rôle peuvent octroyer eux-mêmes n'importe quelle autorisation. En général, n'octroyez pas d'autorisations IAM à des utilisateurs ou des rôles auxquels vous n'approuvez pas l'accès complet aux ressources de votre compte. Lorsque vous effectuez votre audit de sécurité, confirmez que les autorisations IAM suivantes sont octroyées à des identités de confiance :
  - `iam:PutGroupPolicy`
  - `iam:PutRolePolicy`
  - `iam:PutUserPolicy`
  - `iam:CreatePolicy`
  - `iam:CreatePolicyVersion`
  - `iam:AttachGroupPolicy`
  - `iam:AttachRolePolicy`
  - `iam:AttachUserPolicy`
- Assurez-vous que les stratégies n'octroient pas d'autorisations aux services que vous n'utilisez pas. Par exemple, si vous utilisez des politiques AWS gérées, assurez-vous que les politiques AWS gérées utilisées dans votre compte concernent les services que vous utilisez réellement. Pour savoir quelles politiques AWS gérées sont utilisées dans votre compte, utilisez l'`GetAccountAuthorizationDetails`API IAM (AWS CLI commande : `aws iam get-account-authorization-details`).
- Si la politique octroie à un utilisateur l'autorisation de lancer une instance Amazon EC2, elle peut également autoriser l'action `iam:PassRole`, mais si c'est le cas, elle doit explicitement répertorier les rôles que l'utilisateur peut transmettre à l'instance Amazon EC2.

- Examinez toutes les valeurs de l'élément `Action` ou `Resource` qui incluent les `*`. Dans la mesure du possible, octroyez l'accès `Allow` aux actions et aux ressources individuelles dont les utilisateurs ont besoin. Toutefois, voici les raisons pour lesquelles il peut être approprié d'utiliser `*` dans une stratégie :
  - La politique est conçue pour octroyer des autorisations au niveau administratif.
  - Le caractère générique est utilisé pour un ensemble d'actions similaires (par exemple, `Describe*`) pour vous faciliter les choses, et vous vous sentez à l'aise avec la liste complète des actions qui sont référencées de cette façon.
  - Le caractère générique est utilisé pour indiquer une classe de ressources ou un chemin d'accès aux ressources (par exemple, `arn:aws:iam::account-id:users/division_abc/*`), et vous vous sentez à l'aise pour octroyer l'accès à toutes les ressources dans cette classe ou ce chemin d'accès.
  - Une action de service ne prend pas en charge les autorisations au niveau des ressources, et le seul choix pour une ressource est `*`.
- Examinez les noms de stratégie pour vérifier qu'ils reflètent la fonction de la stratégie. Par exemple, même si une stratégie peut avoir un nom qui inclut « lecture seule », la stratégie peut en fait octroyer des autorisations de lecture ou de modification.

Pour de plus amples informations sur la planification de votre audit de sécurité, consultez Meilleures pratiques de sécurité, d'identité et de conformité dans le Centre d'architecture AWS .

Protection des données dans AWS Identity and Access Management

Le modèle de responsabilité AWS partagée de s'applique à la protection des données dans AWS Identity and Access Management. Comme décrit dans ce modèle, AWS est chargé de protéger l'infrastructure mondiale qui gère tous les AWS Cloud. La gestion du contrôle de votre contenu hébergé sur cette infrastructure relève de votre responsabilité. Vous êtes également responsable des tâches de configuration et de gestion de la sécurité des Services AWS que vous utilisez. Pour plus d'informations sur la confidentialité des données, consultez Questions fréquentes (FAQ) sur la confidentialité des données. Pour en savoir plus sur la protection des données en Europe, consultez le billet de blog Modèle de responsabilité partagée AWS et RGPD (Règlement général sur la protection des données) sur le Blog de sécuritéAWS .

À des fins de protection des données, nous vous recommandons de protéger les Compte AWS informations d'identification et de configurer les utilisateurs individuels avec AWS IAM Identity Center ou AWS Identity and Access Management (IAM). Ainsi, chaque utilisateur se voit attribuer uniquement les autorisations nécessaires pour exécuter ses tâches. Nous vous recommandons également de sécuriser vos données comme indiqué ci-dessous :

- Utilisez l'authentification multifactorielle (MFA) avec chaque compte.
- Utilisez le protocole SSL/TLS pour communiquer avec les ressources. AWS Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Configurez l'API et la journalisation de l'activité des utilisateurs avec AWS CloudTrail.
- Utilisez des solutions de AWS chiffrement, ainsi que tous les contrôles de sécurité par défaut qu'ils contiennent Services AWS.
- Utilisez des services de sécurité gérés avancés tels qu'Amazon Macie, qui contribuent à la découverte et à la sécurisation des données sensibles stockées dans Amazon S3.
- Si vous avez besoin de modules cryptographiques validés par la norme FIPS 140-2 pour accéder AWS via une interface de ligne de commande ou une API, utilisez un point de terminaison FIPS. Pour plus d'informations sur les points de terminaison FIPS (Federal Information Processing Standard) disponibles, consultez Federal Information Processing Standard (FIPS) 140-2 (Normes de traitement de l'information fédérale).

Nous vous recommandons fortement de ne jamais placer d'informations confidentielles ou sensibles, telles que les adresses e-mail de vos clients, dans des balises ou des champs de texte libre tels que le champ Name (Nom). Cela inclut lorsque vous travaillez avec IAM ou un autre utilisateur à Services AWS l'aide de la console, de l'API ou des AWS SDK. AWS CLI Toutes les données que vous entrez dans des balises ou des champs de texte de forme libre utilisés pour les noms peuvent être utilisées à des fins de facturation ou dans les journaux de diagnostic. Si vous fournissez une adresse URL à un serveur externe, nous vous recommandons fortement de ne pas inclure d'informations d'identification dans l'adresse URL permettant de valider votre demande adressée à ce serveur.

Chiffrement des données dans IAM et AWS STS

Le chiffrement des données se divise généralement en deux catégories : le chiffrement au repos et le chiffrement en transit.

Chiffrement au repos

Les données collectées et stockées par IAM sont chiffrées au repos.

- IAM : les données collectées et stockées dans IAM comprennent les adresses IP, les métadonnées du compte client et les données d'identification des clients, y compris les mots de passe. Les métadonnées du compte client et les données d'identification des clients sont chiffrées au repos à l'aide d'AES 256 ou sont hachés à l'aide de SHA 256.
- AWS STS— AWS STS ne collecte pas le contenu des clients, à l'exception des journaux de service qui enregistrent les demandes réussies, erronées ou erronées adressées au service.

Chiffrement en transit

Les données d'identification des clients, y compris les mots de passe, sont chiffrées en transit à l'aide des protocoles TLS 1.2 et 1.3. Tous les AWS STS terminaux prennent en charge le protocole HTTPS pour chiffrer les données en transit. Pour obtenir la liste des AWS STS points de terminaison, consultezRégions et points de terminaison.

Gestion des clés dans IAM et AWS STS

Vous ne pouvez pas gérer les clés de chiffrement à l'aide d'IAM ou AWS STS. Pour plus d'informations sur les clés de chiffrement, voir Qu'est-ce que c'est AWS KMS ? dans le guide AWS Key Management Service du développeur

Confidentialité du trafic interréseau dans IAM et AWS STS

Les demandes IAM doivent être effectuées à l'aide du protocole TLS (Transport Layer Security). Vous pouvez sécuriser les connexions au AWS STS service en utilisant des points de terminaison VPC. Pour en savoir plus, consultez Points de terminaison de VPC d'Interface.

Connexion et surveillance AWS Identity and Access Management

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances de AWS Identity and Access Management (IAM), AWS Security Token Service (AWS STS) et de vos autres AWS solutions. AWS fournit plusieurs outils pour surveiller vos AWS ressources et répondre aux incidents potentiels :

- AWS CloudTrailcapture tous les appels d'API pour IAM et AWS STS en tant qu'événements, y compris les appels depuis la console et les appels d'API. Pour en savoir plus sur l'utilisation CloudTrail avec IAM et AWS STS, voirJournalisation des appels IAM et AWS STS API avec AWS CloudTrail. Pour plus d'informations CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.

- AWS Identity and Access Management Access Analyzer vous aide à identifier les ressources de votre organisation et les comptes, tels que les compartiments Amazon S3 ou les rôles IAM, qui sont partagés avec une entité externe. Cela vous aide à identifier les accès imprévus à vos ressources et données, ce qui constitue un risque de sécurité. Pour en savoir plus, consultez Qu'est-ce qu'IAM Access Analyzer ?.
- Amazon CloudWatch surveille vos AWS ressources et les applications que vous utilisez AWS en temps réel. Vous pouvez collecter et suivre les métriques, créer des tableaux de bord personnalisés, et définir des alarmes qui vous informent ou prennent des mesures lorsqu'une métrique spécifique atteint un seuil que vous spécifiez. Par exemple, vous pouvez CloudWatch suivre l'utilisation du processeur ou d'autres indicateurs de vos instances Amazon EC2 et lancer automatiquement de nouvelles instances en cas de besoin. Pour plus d'informations, consultez le guide de CloudWatch l'utilisateur Amazon.
- Amazon CloudWatch Logs vous permet de surveiller, de stocker et d'accéder à vos fichiers journaux à partir d'instances Amazon EC2 et d'autres sources. CloudTrail CloudWatch Les journaux peuvent surveiller les informations contenues dans les fichiers journaux et vous avertir lorsque certains seuils sont atteints. Vous pouvez également archiver vos données de journaux dans une solution de stockage hautement durable. Pour plus d'informations, consultez le guide de l'utilisateur d'Amazon CloudWatch Logs.

Pour consulter des ressources supplémentaires et les bonnes pratiques en matière de sécurité pour IAM, reportez-vous à la section Bonnes pratiques de sécurité et cas d'utilisation dans AWS Identity and Access Management.

Validation de la conformité pour AWS Identity and Access Management

Des auditeurs tiers évaluent la sécurité et la conformité de AWS Identity and Access Management (IAM) dans le cadre de multiples programmes de AWS conformité. Il s'agit notamment des certifications SOC, PCI, FedRAMP, ISO et autres.

Pour savoir si un programme Services AWS de conformité Service AWS s'inscrit dans le champ d'application de programmes de conformité spécifiques, consultez Services AWS la section de conformité et sélectionnez le programme de conformité qui vous intéresse. Pour des informations générales, voir Programmes de AWS conformité Programmes AWS de .

Vous pouvez télécharger des rapports d'audit tiers à l'aide de AWS Artifact. Pour plus d'informations, voir Téléchargement de rapports dans AWS Artifact .

Votre responsabilité en matière de conformité lors de l'utilisation Services AWS est déterminée par la sensibilité de vos données, les objectifs de conformité de votre entreprise et les lois et réglementations applicables. AWS fournit les ressources suivantes pour faciliter la mise en conformité :

- Guides de démarrage rapide sur la sécurité et la conformité : ces guides de déploiement abordent les considérations architecturales et indiquent les étapes à suivre pour déployer des environnements de base axés sur AWS la sécurité et la conformité.
- Architecture axée sur la sécurité et la conformité HIPAA sur Amazon Web Services : ce livre blanc décrit comment les entreprises peuvent créer des applications AWS conformes à la loi HIPAA.

> **Note**
>
> Tous ne Services AWS sont pas éligibles à la loi HIPAA. Pour plus d'informations, consultez le HIPAA Eligible Services Reference.

- AWS Ressources de https://aws.amazon.com/compliance/resources/ de conformité — Cette collection de classeurs et de guides peut s'appliquer à votre secteur d'activité et à votre région.
- AWS Guides de conformité destinés aux clients — Comprenez le modèle de responsabilité partagée sous l'angle de la conformité. Les guides résument les meilleures pratiques en matière de sécurisation Services AWS et décrivent les directives relatives aux contrôles de sécurité dans de nombreux cadres (notamment le National Institute of Standards and Technology (NIST), le Payment Card Industry Security Standards Council (PCI) et l'Organisation internationale de normalisation (ISO)).
- Évaluation des ressources à l'aide des règles du guide du AWS Config développeur : le AWS Config service évalue dans quelle mesure les configurations de vos ressources sont conformes aux pratiques internes, aux directives du secteur et aux réglementations.
- AWS Security Hub— Cela Service AWS fournit une vue complète de votre état de sécurité interne AWS. Security Hub utilise des contrôles de sécurité pour évaluer vos ressources AWS et vérifier votre conformité par rapport aux normes et aux bonnes pratiques du secteur de la sécurité. Pour obtenir la liste des services et des contrôles pris en charge, consultez Référence des contrôles Security Hub.
- Amazon GuardDuty — Cela Service AWS détecte les menaces potentielles qui pèsent sur vos charges de travail Comptes AWS, vos conteneurs et vos données en surveillant votre

environnement pour détecter toute activité suspecte et malveillante. GuardDuty peut vous aider à répondre à diverses exigences de conformité, telles que la norme PCI DSS, en répondant aux exigences de détection des intrusions imposées par certains cadres de conformité.

- AWS Audit Manager— Cela vous Service AWS permet d'auditer en permanence votre AWS utilisation afin de simplifier la gestion des risques et la conformité aux réglementations et aux normes du secteur.

Résilience dans AWS Identity and Access Management

L'infrastructure AWS mondiale est construite autour des AWS régions et des zones de disponibilité. AWS Les régions disposent de plusieurs zones de disponibilité physiquement séparées et isolées, connectées par un réseau à faible latence, à haut débit et hautement redondant. Pour plus d'informations sur AWS les régions et les zones de disponibilité, consultez la section Infrastructure AWS mondiale.

AWS Identity and Access Management (IAM) et AWS Security Token Service (AWS STS) sont des services régionaux autonomes disponibles dans le monde entier.

L'IAM est un élément essentiel Service AWS. Chaque opération effectuée AWS doit être authentifiée et autorisée par IAM. IAM vérifie chaque demande en fonction des identités et des politiques stockées dans IAM pour déterminer si la demande est autorisée ou refusée. IAM a été conçu avec un plan de contrôle et un plan de données séparés afin que le service s'authentifie même en cas de défaillance inattendue. Les ressources IAM utilisées dans les autorisations, telles que les rôles et les politiques, sont stockées dans le plan de contrôle. Les clients IAM peuvent modifier la configuration de ces ressources en utilisant des opérations IAM telles que `DeletePolicy` et `AttachRolePolicy`. Ces demandes de modification de configuration sont envoyées au plan de contrôle. Il existe un seul plan de contrôle IAM pour tous les avions commerciaux Régions AWS, situé dans la région de l'est des États-Unis (Virginie du Nord). Le système IAM propage ensuite les modifications de configuration aux plans de données IAM dans chaque Région AWS activée. Le plan de données IAM est essentiellement une réplique en lecture seule des données de configuration du plan de contrôle IAM. Chacun Région AWS dispose d'une instance totalement indépendante du plan de données IAM, qui effectue l'authentification et l'autorisation pour les demandes provenant de la même région. Dans chaque région, le plan de données IAM est réparti sur au moins trois zones de disponibilité et possède une capacité suffisante pour tolérer la perte d'une zone de disponibilité sans nuire au client. Les plans de contrôle et de données IAM ont été conçus pour un temps d'interruption planifié nul, toutes les mises à jour logicielles et les opérations de mise à l'échelle étant effectuées de manière invisible pour les clients.

AWS STS les demandes sont toujours dirigées vers un seul point de terminaison global par défaut. Vous pouvez utiliser un point de terminaison AWS STS régional pour réduire la latence ou fournir une redondance supplémentaire pour vos applications. Pour en savoir plus, veuillez consulter la section Gérer AWS STS dans un Région AWS.

Certains événements peuvent interrompre la communication Régions AWS entre les réseaux. Cependant, même lorsque vous ne pouvez pas communiquer avec le point de terminaison IAM global, vous AWS STS pouvez toujours authentifier les principaux IAM et IAM peut autoriser vos demandes. Les détails spécifiques d'un événement qui interrompt la communication détermineront votre capacité à accéder aux AWS services. Dans la plupart des cas, vous pouvez continuer à utiliser les informations d'identification IAM dans votre AWS environnement. Les conditions suivantes peuvent s'appliquer à un événement qui interrompt la communication.

Clés d'accès pour utilisateurs IAM

Vous pouvez vous authentifier indéfiniment dans une région avec les clés d'accès longue durée pour utilisateurs IAM. Lorsque vous utilisez les API AWS Command Line Interface et, vous pouvez fournir des clés AWS d'accès afin AWS de vérifier votre identité dans les demandes programmatiques.

> **Important**
>
> Selon les bonnes pratiques, nous recommandons à vos utilisateurs de se connecter avec des informations d'identification temporaires plutôt qu'avec des clés d'accès longue durée.

Informations d’identification temporaires

Vous pouvez demander de nouvelles informations d'identification temporaires auprès du point de terminaison du service AWS STS régional pendant au moins 24 heures. Les opérations d'API suivantes génèrent des informations d'identification temporaires.

- AssumeRole
- AssumeRoleWithWebIdentity
- AssumeRoleWithSAML
- GetFederationToken
- GetSessionToken

Principaux et autorisations

- Il se peut que vous ne puissiez pas ajouter, modifier ou supprimer des principaux ou des autorisations dans IAM.
- Il se peut que vos informations d'identification ne reflètent pas les modifications que vous avez récemment apportées à vos autorisations dans IAM. Pour plus d'informations, consultez Les modifications que j'apporte ne sont pas toujours visibles immédiatement.

AWS Management Console

- Vous pouvez utiliser un point de terminaison de connexion régional pour vous connecter à l' AWS Management Console en tant qu'utilisateur IAM. Les points de terminaison de connexion régionaux ont le format d'URL suivant.

  `https://{Account ID}.signin.aws.amazon.com/console?region={Region}`

  Exemple : https://111122223333.signin.aws.amazon.com/console?region=us-west-2
- Il se peut que vous ne puissiez pas terminer l'authentification multifactorielle (MFA) Universal 2nd Factor (U2F).

Bonnes pratiques pour la résilience IAM

AWS a intégré la résilience dans Régions AWS les zones de disponibilité. Lorsque vous respectez les bonnes pratiques IAM suivantes dans les systèmes qui interagissent avec votre environnement, vous profitez de cette résilience.

1. Utilisez un point de terminaison de service AWS STS régional au lieu du point de terminaison global par défaut.
2. Passez en revue la configuration de votre environnement pour les ressources vitales qui créent ou modifient régulièrement des ressources IAM, et préparez une solution de secours qui utilise les ressources IAM existantes.

Sécurité de l'infrastructure dans AWS Identity and Access Management

En tant que service géré, AWS Identity and Access Management est protégé par la sécurité du réseau AWS mondial. Pour plus d'informations sur les services AWS de sécurité et sur la manière dont AWS l'infrastructure est protégée, consultez la section Sécurité du AWS cloud. Pour

concevoir votre AWS environnement en utilisant les meilleures pratiques en matière de sécurité de l'infrastructure, consultez la section Protection de l'infrastructure dans le cadre AWS bien architecturé du pilier de sécurité.

Vous utilisez des appels d'API AWS publiés pour accéder à IAM via le réseau. Les clients doivent prendre en charge les éléments suivants :

- Protocole TLS (Transport Layer Security). Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Ses suites de chiffrement PFS (Perfect Forward Secrecy) comme DHE (Ephemeral Diffie-Hellman) ou ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

En outre, les demandes doivent être signées à l'aide d'un ID de clé d'accès et d'une clé d'accès secrète associée à un principal IAM. Vous pouvez également utiliser AWS Security Token Service (AWS STS) pour générer des informations d'identification de sécurité temporaires et signer les demandes.

IAM est accessible par programmation à l'aide de l'API HTTPS IAM, qui vous permet d'effectuer des demandes HTTPS directement au service. L'API Query renvoie des informations sensibles, y compris des informations d'identification de sécurité. Par conséquent, vous devez utiliser HTTPS avec toutes les demandes d'API. Lorsque vous utilisez l'API HTTPS, vous devez inclure du code pour signer numériquement les demandes à l'aide de vos informations d'identification.

Vous pouvez appeler ces opérations d'API à partir de n'importe quel emplacement sur le réseau, mais IAM prend en charge les politiques d'accès basées sur les ressources, ce qui peut inclure des restrictions en fonction de l'adresse IP source. Vous pouvez également utiliser des politiques IAM pour contrôler l'accès à partir de points de terminaison Amazon Virtual Private Cloud (Amazon VPC) ou de VPC spécifiques. En fait, cela isole l'accès réseau à une ressource IAM donnée uniquement du VPC spécifique au sein du réseau. AWS

Analyse de configuration et de vulnérabilité dans AWS Identity and Access Management

AWS gère les tâches de sécurité de base telles que l'application de correctifs au système d'exploitation client (OS) et aux bases de données, la configuration du pare-feu et la reprise après sinistre. Ces procédures ont été vérifiées et certifiées par les tiers appropriés. Pour plus de détails, consultez les ressources suivantes :

- Modèle de responsabilité partagée
- Amazon Web Services : Présentation des procédures de sécurité (livre blanc)

Les ressources suivantes traitent également de la configuration et de l'analyse des vulnérabilités dans AWS Identity and Access Management (IAM) :

- Validation de la conformité pour AWS Identity and Access Management
- Bonnes pratiques de sécurité et cas d'utilisation dans AWS Identity and Access Management

AWS politiques gérées pour AWS Identity and Access Management Access Analyzer

Une politique AWS gérée est une politique autonome créée et administrée par AWS. AWS les politiques gérées sont conçues pour fournir des autorisations pour de nombreux cas d'utilisation courants afin que vous puissiez commencer à attribuer des autorisations aux utilisateurs, aux groupes et aux rôles.

N'oubliez pas que les politiques AWS gérées peuvent ne pas accorder d'autorisations de moindre privilège pour vos cas d'utilisation spécifiques, car elles sont accessibles à tous les AWS clients. Nous vous recommandons de réduire encore les autorisations en définissant des politiques gérées par le client qui sont propres à vos cas d’utilisation.

Vous ne pouvez pas modifier les autorisations définies dans les politiques AWS gérées. Si les autorisations définies dans une politique AWS gérée sont AWS mises à jour, la mise à jour affecte toutes les identités principales (utilisateurs, groupes et rôles) auxquelles la politique est attachée. AWS est le plus susceptible de mettre à jour une politique AWS gérée lorsqu'une nouvelle politique Service AWS est lancée ou lorsque de nouvelles opérations d'API sont disponibles pour les services existants.

Pour plus d’informations, consultez la section Politiques gérées par AWS dans le Guide de l’utilisateur IAM.

Accès IAM ReadOnly

Utilisez la politique gérée `IAMReadOnlyAccess` pour autoriser l'accès en lecture seule aux ressources IAM. Cette politique accorde l'autorisation d'obtenir et de répertorier toutes les ressources IAM. Elle permet d'afficher les détails et les rapports d'activité pour les utilisateurs, les groupes, les rôles, les politiques, les fournisseurs d'identité et les dispositifs MFA. Elle n'inclut pas la possibilité de créer ou de supprimer des ressources ou d'accéder aux ressources IAM Access Analyzer. Consultez la politique pour obtenir la liste complète des services et des actions que prend en charge cette politique.

Mot de passe IAM UserChange

Utilisez la politique gérée par l'interface `IAMUserChangePassword` pour autoriser les utilisateurs IAM à modifier leur mot de passe.

Vous configurez vos Account settings (paramètres de compte) IAM et la Password policy (politique de mot de passe) pour autoriser les utilisateurs IAM à modifier le mot de passe de leur compte IAM. Lorsque vous autorisez cette action, IAM attache la politique suivante à chaque utilisateur :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

IAM AccessAnalyzer FullAccess

Utilisez la politique `IAMAccessAnalyzerFullAccess` AWS gérée pour permettre à vos administrateurs d'accéder à IAM Access Analyzer.

Groupes d'autorisations

Cette politique est groupée en instructions basées sur le jeu d'autorisations fourni.

- IAM Access Analyzer : octroie des autorisations administratives totales à l'ensemble des ressources dans IAM Access Analyzer.
- Créer un rôle lié au service : autorise l'administrateur à créer un rôle lié au service qui autorise IAM Access Analyzer à analyser les ressources dans d'autres services en votre nom. Cette autorisation autorise la création du rôle lié au service uniquement pour une utilisation par IAM Access Analyzer.
- AWS Organizations : autorise les administrateurs à utiliser IAM Access Analyzer pour une organisation dans AWS Organizations. Après avoir activé l'accès sécurisé pour IAM Access Analyzer dans AWS Organizations, les membres du compte de gestion peuvent consulter les résultats au sein de leur organisation.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "access-analyzer:*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "access-analyzer.amazonaws.com"
        }
      }
    },
    {
```

```
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListChildren",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListParents",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

Accès IAM AccessAnalyzer ReadOnly

Utilisez la politique `IAMAccessAnalyzerReadOnlyAccess` AWS gérée pour autoriser l'accès en lecture seule à IAM Access Analyzer.

Pour autoriser également l'accès en lecture seule à IAM Access Analyzer pour AWS Organizations, créez une politique gérée par le client qui autorise les actions de description et de liste à partir de la politique gérée. IAM AccessAnalyzer FullAccess AWS

Autorisations de niveau service

Cette politique fournit un accès en lecture seule à IAM Access Analyzer. Cette politique n'inclut aucune autre autorisation de service.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMAccessAnalyzerReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "access-analyzer:CheckAccessNotGranted",
        "access-analyzer:CheckNoNewAccess",
```

```
        "access-analyzer:Get*",
        "access-analyzer:List*",
        "access-analyzer:ValidatePolicy"
      ],
      "Resource": "*"
    }
  ]
}
```

AccessAnalyzerServiceRolePolitique

Vous ne pouvez pas vous associer AccessAnalyzerServiceRolePolicy à vos entités IAM. Cette politique est attachée à un rôle lié à un service qui permet à l'analyseur d'accès IAM de réaliser des actions en votre nom. Pour plus d'informations, consultez la section Utilisation de rôles liés à un service pour AWS Identity and Access Management Access Analyzer.

Groupes d'autorisations

Cette politique autorise l'accès à IAM Access Analyzer pour analyser les métadonnées des ressources à partir de plusieurs Services AWS.

- Amazon DynamoDB : autorise l'affichage des flux et des tables DynamoDB.
- Amazon Elastic Compute Cloud : autorise la description des adresses IP, des instantanés et des VPC.
- Amazon Elastic Container Registry : autorise la description des référentiels d'images et les politiques de récupération des référentiels.
- Amazon Elastic File System : autorise l'affichage de la description d'un système de fichiers Amazon EFS et de la politique de niveau de ressources d'un système de fichiers Amazon EFS.
- AWS Identity and Access Management - Permet de récupérer des informations sur un rôle spécifié et de répertorier les rôles IAM dotés d'un préfixe de chemin spécifié. Permet de récupérer des informations sur les utilisateurs, les groupes d'utilisateurs, les profils de connexion, les clés d'accès et les dernières données du service consultées.
- AWS Key Management Service - Autorise l'affichage d'informations détaillées sur une clé KMS, ses politiques clés et ses subventions.
- AWS Lambda - Autorise l'affichage des informations sur les alias, fonctions, couches et alias Lambda.
- AWS Organizations— Autorise les autorisations aux Organizations et permet la création d'un analyseur au sein de l' AWS organisation en tant que zone de confiance.

- Amazon Relational Database Service : autorise l'affichage d'informations détaillées sur les instantanés de base de données Amazon RDS et les instantanés de clusters de bases de données Amazon RDS.
- Amazon Simple Storage Service : autorise l'accès à des informations détaillées sur les points d'accès Amazon S3, les compartiments et les compartiments de répertoire Amazon S3 qui utilisent la classe de stockage Amazon S3 Express One.
- AWS Secrets Manager - Autorise l'affichage d'informations détaillées sur les secrets et les politiques en matière de ressources associées aux secrets.
- Amazon Simple Notification Service : autorise l'affichage d'informations détaillées sur un sujet.
- Amazon Simple Queue Service : autorise l'affichage d'informations détaillées sur les files d'attente spécifiées.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccessAnalyzerServiceRolePolicy",
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetResourcePolicy",
        "dynamodb:ListStreams",
        "dynamodb:ListTables",
        "ec2:DescribeAddresses",
        "ec2:DescribeByoipCidrs",
        "ec2:DescribeSnapshotAttribute",
        "ec2:DescribeSnapshots",
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeVpcs",
        "ec2:GetSnapshotBlockPublicAccessState",
        "ecr:DescribeRepositories",
        "ecr:GetRepositoryPolicy",
        "elasticfilesystem:DescribeFileSystemPolicy",
        "elasticfilesystem:DescribeFileSystems",
        "iam:GetRole",
        "iam:ListEntitiesForPolicy",
        "iam:ListRoles",
        "iam:ListUsers",
        "iam:GetUser",
        "iam:GetGroup",
        "iam:GenerateServiceLastAccessedDetails",
```

```
        "iam:GetServiceLastAccessedDetails",
        "iam:ListAccessKeys",
        "iam:GetLoginProfile",
        "iam:GetAccessKeyLastUsed",
        "iam:ListRolePolicies",
        "iam:GetRolePolicy",
        "iam:ListAttachedRolePolicies",
        "iam:ListUserPolicies",
        "iam:GetUserPolicy",
        "iam:ListAttachedUserPolicies",
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:ListGroupsForUser",
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms:ListGrants",
        "kms:ListKeyPolicies",
        "kms:ListKeys",
        "lambda:GetFunctionUrlConfig",
        "lambda:GetLayerVersionPolicy",
        "lambda:GetPolicy",
        "lambda:ListAliases",
        "lambda:ListFunctions",
        "lambda:ListLayers",
        "lambda:ListLayerVersions",
        "lambda:ListVersionsByFunction",
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListChildren",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListParents",
        "organizations:ListRoots",
        "rds:DescribeDBClusterSnapshotAttributes",
        "rds:DescribeDBClusterSnapshots",
        "rds:DescribeDBSnapshotAttributes",
        "rds:DescribeDBSnapshots",
        "s3:DescribeMultiRegionAccessPointOperation",
        "s3:GetAccessPoint",
        "s3:GetAccessPointPolicy",
```

```
        "s3:GetAccessPointPolicyStatus",
        "s3:GetAccountPublicAccessBlock",
        "s3:GetBucketAcl",
        "s3:GetBucketLocation",
        "s3:GetBucketPolicyStatus",
        "s3:GetBucketPolicy",
        "s3:GetBucketPublicAccessBlock",
        "s3:GetMultiRegionAccessPoint",
        "s3:GetMultiRegionAccessPointPolicy",
        "s3:GetMultiRegionAccessPointPolicyStatus",
        "s3:ListAccessPoints",
        "s3:ListAllMyBuckets",
        "s3:ListMultiRegionAccessPoints",
        "s3express:GetBucketPolicy",
        "s3express:ListAllMyDirectoryBuckets",
        "sns:GetTopicAttributes",
        "sns:ListTopics",
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetResourcePolicy",
        "secretsmanager:ListSecrets",
        "sqs:GetQueueAttributes",
        "sqs:ListQueues"
      ],
      "Resource": "*"
    }
  ]
}
```

Mises à jour de IAM et de IAM Access Analyzer pour les politiques gérées par l'interface AWS

Consultez les informations relatives aux mises à jour apportées à l'IAM et aux politiques AWS gérées depuis que le service a commencé à suivre ces modifications. Pour recevoir des alertes automatiques sur les modifications apportées à cette page, abonnez-vous au flux RSS sur les pages d'historique d'IAM et IAM Access Analyzer Document.

Modification	Description	Date
AccessAnalyzerServiceRolePolitique — Autorisations ajoutées	IAM Access Analyzer a ajouté la prise en charge de l'autorisation de récupérer des informations sur les politiques relatives aux utilisateurs et aux rôles IAM aux autorisations de niveau de service de. `AccessAnalyzerServiceRolePolicy`	30 mai 2024
AccessAnalyzerServiceRolePolitique — Autorisations ajoutées	IAM Access Analyzer a ajouté la prise en charge de l'autorisation de récupérer l'état actuel du blocage de l'accès public pour les instantanés Amazon EC2 aux autorisations de niveau de service de. `AccessAnalyzerServiceRolePolicy`	23 janvier 2024
AccessAnalyzerServiceRolePolitique — Autorisations ajoutées	IAM Access Analyzer a ajouté la prise en charge des flux et des tables DynamoDB aux autorisations de niveau de service de. `AccessAnalyzerServiceRolePolicy`	11 janvier 2024
AccessAnalyzerServiceRolePolitique — Autorisations ajoutées	IAM Access Analyzer a ajouté la prise en charge des compartiments d'annuaire Amazon S3 aux autorisations de niveau de service de. `AccessAnalyzerServiceRolePolicy`	1er décembre 2023

Modification	Description	Date
AccessAnalyzerReadOnlyAccès IAM — Autorisations ajoutées	L’analyseur d’accès IAM a ajouté des autorisations pour vous permettre de vérifier si les mises à jour de vos politiques accordent un accès supplémentaire. L'analyseur d'accès IAM a besoin de cette autorisation pour effectuer des vérifications de politique sur vos politiques.	26 novembre 2023
AccessAnalyzerServiceRolePolitique — Autorisations ajoutées	L’analyseur d’accès IAM a ajouté des actions IAM aux autorisations de niveau du service d’`AccessAnalyzerServiceRolePolicy` pour prendre en charge les actions suivantes : • Lister les entités pour une politique • Générer les dernières informations consultées sur le service • Lister les informations clés d’accès	26 novembre 2023

Modification	Description	Date
AccessAnalyzerServiceRolePolitique — Autorisations ajoutées	IAM Access Analyzer a ajouté la prise en charge des types de ressources suivants sur les autorisations de niveau de service de `AccessAnalyzerServiceRolePolicy` : • Instantanés de volumes Amazon EBS • Référentiels Amazon ECR • Système de fichiers Amazon EFS • Instantanés de base de données Amazon RDS • Instantanés du cluster de base de données Amazon RDS • Rubriques Amazon SNS	25 octobre 2022
AccessAnalyzerServiceRolePolitique — Autorisations ajoutées	L'analyseur d'accès IAM a ajouté l'action `lambda:GetFunctionUrlConfig` sur les autorisations de niveau de service de `AccessAnalyzerServiceRolePolicy` .	6 avril 2022
AccessAnalyzerServiceRolePolitique — Autorisations ajoutées	L'analyseur d'accès IAM a ajouté de nouvelles actions Amazon S3 pour analyser les métadonnées associées aux points d'accès multi-régions.	2 septembre 2021

Modification	Description	Date
AccessAnalyzerReadOnlyAccès IAM — Autorisations ajoutées	L'analyseur d'accès IAM a ajouté une nouvelle action pour octroyer des autorisations `ValidatePolicy` visant à vous autoriser à utiliser les vérifications de politique pour la validation. L'analyseur d'accès IAM a besoin de cette autorisation pour effectuer des vérifications de politique sur vos politiques.	16 mars 2021
IAM Access Analyzer a commencé à suivre les modifications	IAM Access Analyzer a commencé à suivre les modifications apportées à ses politiques AWS gérées.	1er mars 2021

En utilisant AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer fournit les fonctionnalités suivantes :

- Les analyseurs d’accès externe de l’analyseur d’accès IAM vous aide à identifier les ressources de votre organisation et de vos comptes qui sont partagés avec une entité externe.
- Les analyseurs d’accès non utilisés de l’analyseur d’accès IAM aident à identifier les accès non utilisés au sein de votre organisation et de vos comptes.
- IAM Access Analyzer valide les politiques IAM par rapport à la grammaire des politiques et aux meilleures pratiques. AWS
- Les vérifications de politiques personnalisées de l’analyseur d’accès IAM permettent de valider les politiques IAM par rapport aux normes de sécurité spécifiées..
- IAM Access Analyzer génère des politiques IAM basées sur l'activité d'accès dans vos journaux. AWS CloudTrail

Identification des ressources partagées avec une entité externe

L’IAM Access Analyzer vous aide à identifier les ressources de votre organisation et de vos comptes, telles que les compartiments Amazon S3 ou les rôles IAM, partagés avec une entité externe. Cela vous permet d'identifier les accès imprévus à vos ressources et données, ce qui constitue un risque de sécurité. IAM Access Analyzer identifie les ressources partagées avec des acteurs externes en utilisant un raisonnement basé sur la logique pour analyser les politiques basées sur les ressources dans votre environnement. AWS Pour chaque instance d'une ressource qui est partagée en dehors de votre compte, l’IAM Access Analyzer génère un résultat. Les résultats comprennent des renseignements sur l'accès et le principal externe à qui il est accordé. Vous pouvez réviser les résultats pour déterminer si l'accès est intentionnel et sûr ou s'il est non intentionnel et représente un risque pour la sécurité. En plus de vous aider à identifier les ressources partagées avec une entité externe, vous pouvez utiliser les résultats de l’IAM Access Analyzer pour prévisualiser la façon dont votre politique affecte l'accès public et les entre comptes à votre ressource avant de déployer des autorisations de ressources. Les résultats sont organisés dans un tableau de bord récapitulatif visuel. Le tableau de bord met en évidence la distinction entre les résultats relatifs à l’accès public et ceux relatifs à l’accès intercomptes, et fournit une ventilation des résultats par type de ressource. Pour

plus d'informations sur les tableaux de bord, consultez Affichage du tableau de bord des résultats de l'analyseur d'accès IAM.

> **Note**
>
> Une entité externe peut être un autre AWS compte, un utilisateur root, un utilisateur ou un rôle IAM, un utilisateur fédéré, un AWS service, un utilisateur anonyme ou une autre entité que vous pouvez utiliser pour créer un filtre. Pour plus d'informations, veuillez consulter Éléments de politique JSON AWS : Principal.

Lorsque vous activez l'IAM Access Analyzer, vous créez un analyseur pour l'ensemble de votre organisation ou de votre compte. L'organisation ou le compte que vous sélectionnez est la zone de confiance de l'analyseur. L'analyseur surveille toutes les ressources prises en charge dans votre zone de confiance. Tout accès aux ressources par les principaux qui se trouvent dans votre zone de confiance est considéré comme fiable. Une fois activé, l'IAM Access Analyzer analyse les politiques appliquées à toutes les ressources prises en charge dans votre zone de confiance. Après la première analyse, l'IAM Access Analyzer analyse lesdites politiques de façon périodique. Si vous ajoutez une nouvelle politique ou modifiez une déja existante, l'IAM Access Analyzer analyse la nouvelle ou celle mise à jour dans un délai d'environ 30 minutes.

Lors de l'analyse des politiques, si l'IAM Access Analyzer en identifie une qui accorde l'accès à un principal externe ne se trouvant pas dans votre zone de confiance, il génère un résultat. Chaque résultat inclut des détails sur la ressource, l'entité externe qui y a accès et les autorisations accordées, afin que vous puissiez prendre les mesures appropriées. Vous pouvez afficher les détails inclus dans le résultat pour déterminer si l'accès à la ressource est intentionnel ou représente un risque potentiel que vous devez résoudre. Lorsque vous ajoutez une politique à une ressource ou que mettez à jour une politique existante, l'IAM Access Analyzer. En outre, l'IAM Access Analyzer analyse périodiquement toutes les politiques basées sur les ressources.

Dans de rares cas, dans certaines conditions, IAM Access Analyzer ne reçoit aucune notification concernant l'ajout ou la mise à jour d'une politique, ce qui peut retarder la génération des résultats. IAM Access Analyzer peut prendre jusqu'à 6 heures pour générer ou résoudre les résultats si vous créez ou supprimez un point d'accès multirégional associé à un compartiment Amazon S3, ou si vous mettez à jour la politique du point d'accès multirégional. En outre, en cas de problème de livraison lié à la livraison du AWS CloudTrail journal, le changement de politique ne déclenche pas une nouvelle analyse de la ressource indiquée dans le résultat. Lorsque cela se produit, l'IAM Access Analyzer analyse la politique nouvelle ou mise à jour au cours de l'analyse périodique suivante, qui a lieu dans

un délai maximal de 24 heures. Si vous souhaitez confirmer qu'une modification apportée à une politique résout un problème d'accès signalé dans un résultat, vous pouvez réanalyser la ressource indiquée dans le résultat à l'aide du Rescan (Réanalyser) lien de la page des détails Findings (Résultats) ou à l'aide de `StartResourceScan` l'opération de l'API de l'IAM Access Analyzer. Pour en savoir plus, veuillez consulter la section Résolution des résultats.

> **Important**
>
> IAM Access Analyzer analyse uniquement les politiques appliquées aux ressources de la même AWS région où il est activé. Pour surveiller toutes les ressources de votre AWS environnement, vous devez créer un analyseur pour activer IAM Access Analyzer dans chaque région où vous utilisez des ressources prises en charge. AWS

L'IAM Access Analyzer prend en charge les types de ressources suivants :

- Compartiments Amazon Simple Storage Service
- Compartiments du répertoire Amazon Simple Storage Service
- AWS Identity and Access Management rôles
- AWS Key Management Service clés
- AWS Lambda fonctions et couches
- Files d'attente Amazon Simple Queue Service
- AWS Secrets Manager secrets
- Rubriques Amazon Simple Notification Service
- Instantanés volumes Amazon Elastic Block Store
- Instantanés de base de données service de base de données relationnelle Amazon
- Instantanés de cluster de base de données service base de données relationnelle Amazon
- Référentiels Amazon Elastic Container Registry
- Systèmes de fichiers Amazon Elastic File System
- Streams Amazon DynamoDB
- Tables Amazon DynamoDB

Identification des accès non utilisés accordés aux utilisateurs et aux rôles IAM

IAM Access Analyzer vous aide à identifier et à examiner les accès non utilisés au sein de votre AWS organisation et de vos comptes. L'analyseur d'accès IAM surveille en permanence tous les rôles et utilisateurs IAM dans votre organisation et vos comptes AWS et génère des résultats pour les accès non utilisés. Les résultats mettent en évidence les rôles non utilisés ainsi que les clés d'accès et les mots de passe non utilisés pour les utilisateurs IAM. Pour les rôles et les utilisateurs IAM actifs, les résultats fournissent une visibilité sur les services et les actions non utilisés.

Les résultats relatifs à la fois aux analyseurs d'accès externes et non utilisés sont organisés dans un tableau de bord récapitulatif visuel. Le tableau de bord met en évidence ceux Comptes AWS qui ont le plus de résultats et fournit une ventilation des résultats par type. Pour de plus amples informations sur le tableau de bord, veuillez consulter Affichage du tableau de bord des résultats de l'analyseur d'accès IAM.

IAM Access Analyzer passe en revue les dernières informations consultées pour tous les rôles et comptes de votre AWS organisation afin de vous aider à identifier les accès non utilisés. Les informations auxquelles vous avez accédé pour la dernière fois sur les actions IAM vous aident à identifier les actions non utilisées pour les rôles de vos Comptes AWS. Pour plus d'informations, consultez Affiner les autorisations en AWS utilisant les dernières informations consultées.

Validation des politiques par rapport aux bonnes pratiques AWS

Vous pouvez valider vos politiques par rapport à la grammaire des politiques IAM et aux bonnes pratiques AWS à l'aide des vérifications de politique de base fournies par la validation des politiques de l'analyseur d'accès IAM. Vous pouvez créer ou modifier une politique à l'aide de l' AWS CLI AWS API ou de l'éditeur de stratégie JSON dans la console IAM. Vous pouvez afficher les résultats des vérifications de validation de politique qui incluent des avertissements de sécurité, des erreurs, des avertissements généraux et des suggestions pour votre politique. Ces résultats fournissent des recommandations exploitables qui vous aident à créer des politiques fonctionnelles et conformes aux AWS meilleures pratiques. Pour en savoir plus sur la validation des politiques à l'aide de la validation des politiques, consultez Validation de la politique de l'IAM Access Analyzer.

Validation des politiques par rapport aux normes de sécurité que vous avez spécifiées

Vous pouvez valider vos politiques par rapport aux normes de sécurité spécifiées à l'aide des vérifications de politiques personnalisées de l'analyseur d'accès IAM. Vous pouvez créer ou modifier une politique à l'aide de l' AWS CLI AWS API ou de l'éditeur de stratégie JSON dans la console IAM. Avec la console, vous pouvez vérifier si votre politique mise à jour accorde un nouvel accès par rapport à la version existante. Grâce à AWS CLI une AWS API, vous pouvez également vérifier que des actions IAM spécifiques que vous considérez comme critiques ne sont pas autorisées par une politique. Ces vérifications mettent en évidence une déclaration de politique qui accorde un nouvel accès. Vous pouvez mettre à jour la déclaration de politique et réexécuter les vérifications jusqu'à ce que la politique soit conforme à votre norme de sécurité. Pour en savoir plus sur la validation des politiques à l'aide de vérifications de politique personnalisées, consultez Vérifications de politiques personnalisées de l'analyseur d'accès IAM.

Générer des politiques

IAM Access Analyzer analyse vos AWS CloudTrail journaux pour identifier les actions et les services qui ont été utilisés par une entité IAM (utilisateur ou rôle) dans la plage de dates spécifiée. Il génère ensuite une politique IAM basée sur cette activité d'accès. Vous pouvez utiliser la politique générée pour affiner les autorisations d'une entité, en l'attachant à un utilisateur ou un rôle IAM. Pour en savoir plus sur la génération de politiques à l'aide de l'IAM Access Analyzer, veuillez consulter Génération d'une politique IAM Access Analyzer.

Tarification pour l'analyseur d'accès IAM

L'analyseur d'accès IAM facture l'analyse des accès non utilisés en fonction du nombre de rôles et d'utilisateurs IAM analysés par analyseur et par mois.

- Vous serez facturé pour chaque analyseur d'accès non utilisé que vous créez.
- Si vous créez des analyseurs d'accès non utilisés dans plusieurs régions, vous serez facturé pour chaque analyseur.
- Les rôles liés à un service ne sont pas analysés pour détecter les activités d'accès non utilisées et ne sont pas inclus dans le nombre total de rôles IAM analysés.

L’analyseur d’accès IAM facture les vérifications de politiques personnalisées en fonction du nombre de demandes d’API faites à l’analyseur d’accès IAM pour vérifier les nouveaux accès.

Pour une liste complète des frais et des prix pour l’analyseur d’accès IAM, consultez la Tarification de l’analyseur d’accès IAM.

Pour consulter votre facture, dirigez-vous vers le Tableau de bord de gestion des coûts et de la facturation dans la console AWS Billing and Cost Management. Votre facture contient des liens vers les rapports d’utilisation qui fournissent des détails sur votre facture. Pour en savoir plus sur la Compte AWS facturation, consultez le guide de AWS Billing l'utilisateur

Si vous avez des questions concernant AWS la facturation, les comptes et les événements, contactez AWS Support.

Résultats des accès externes et non utilisés

L’analyseur d’accès IAM génère des résultats des accès externes et des accès non utilisés dans votre Compte AWS ou votre organisation. Pour l’accès externe, l’analyseur d’accès IAM génère un résultat pour chaque instance de politique basée sur les ressources accordant l’accès à une ressource dans votre zone de confiance à un principal qui ne se trouve dans votre zone de confiance. Lorsque vous créez un analyseur d'accès externe, vous choisissez une organisation ou vous souhaitez l' Compte AWS analyser. Tout principal de l'organisation ou du compte que vous sélectionnez pour l'analyseur est considéré comme approuvé. Étant donné que les principaux de la même organisation ou du même compte sont approuvés, les ressources et les principaux de l'organisation ou du compte constituent la zone de confiance de l'analyseur. Tout partage se trouvant dans la zone de confiance est considéré comme sûr. L’IAM Access Analyzer ne génère donc pas de résultat. Par exemple, si vous sélectionnez une organisation comme zone de confiance d'un analyseur, toutes les ressources et tous les principaux de cette organisation sont compris dans cette zone de confiance. Si vous accordez des autorisations à un compartiment Amazon S3 dans l’un des comptes membres de votre organisation à un principal dans un autre compte membre de celle-ci, l’analyseur d’accès IAM ne génère pas de résultat. Toutefois, si vous accordez une autorisation à un principal dans un compte qui n'est pas membre de l'organisation, l’IAM Access Analyzer génère un résultat.

IAM Access Analyzer génère également des résultats concernant les accès non utilisés accordés dans votre AWS organisation et dans vos comptes. Lorsque vous créez un analyseur d'accès non utilisé, IAM Access Analyzer surveille en permanence tous les rôles et utilisateurs IAM de votre AWS organisation et de vos comptes et génère des résultats pour les accès non utilisés. L’analyseur d’accès IAM génère les types de résultats suivants pour les accès non utilisés :

- Rôles non utilisés : rôles sans activité d'accès dans la fenêtre d'utilisation spécifiée.
- Clés d'accès et mots de passe utilisateur IAM non utilisés : informations d'identification appartenant aux utilisateurs IAM qui leur permettent d'accéder à votre Compte AWS.
- Autorisations non utilisées : autorisations au niveau du service et de l'action qui n'ont pas été utilisées par un rôle dans la fenêtre d'utilisation spécifiée. L'analyseur d'accès IAM utilise des politiques basées sur l'identité attachées aux rôles pour déterminer les services et les actions auxquels ces rôles peuvent accéder. L'analyseur d'accès IAM prend en charge l'examen des autorisations non utilisées pour toutes les autorisations de niveau du service. Pour obtenir la liste complète des autorisations au niveau de l'action prises en charge pour les résultats des accès non utilisés, consultez Services et actions concernant les dernières informations consultées relatives à une action IAM.

> **Note**
>
> L'analyseur d'accès IAM fournit des résultats des accès externes gratuitement et facture les résultats des accès non utilisés en fonction du nombre de rôles et d'utilisateurs IAM analysés par analyseur et par mois. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

Rubriques

- Comment fonctionnent les résultats de l'IAM Access Analyzer
- Démarrage avec AWS Identity and Access Management Access Analyzer résultats
- Affichage du tableau de bord des résultats de l'analyseur d'accès IAM
- Utilisation des résultats
- Examen des résultats
- Filtrage des résultats
- Archivage des résultats
- Résolution des résultats
- Types de ressources de l'analyseur d'accès IAM pour l'accès externe
- Paramètres pour IAM Access Analyzer
- Règles d'archivage
- Surveillance AWS Identity and Access Management Access Analyzer avec Amazon EventBridge

- Intégrez Access Analyzer à AWS Security Hub
- Journalisation des appels d'API IAM Access Analyzer avec AWS CloudTrail
- Clés de filtre de l’IAM Access Analyzer
- Utilisation des rôles liés aux services pour AWS Identity and Access Management Access Analyzer

Comment fonctionnent les résultats de l’IAM Access Analyzer

Cette rubrique décrit les concepts et termes utilisés dans IAM Access Analyzer pour vous aider à vous familiariser avec la manière dont IAM Access Analyzer surveille l'accès à vos ressources. AWS

Accès externe

Pour les analyseurs d'accès externes, AWS Identity and Access Management Access Analyzer il est basé sur Zelkova, qui traduit les politiques IAM en instructions logiques équivalentes et exécute une suite de solveurs logiques spécialisés et à usage général (théories du modulo de satisfaisabilité) pour résoudre le problème. l’IAM Access Analyzer applique Zelkova de manière répétitive à une politique avec des requêtes de plus en plus spécifiques pour caractériser les classes de comportements autorisées par la politique, en fonction du contenu de celle-ci Pour en savoir plus sur la satisfiabilité modulo des théories, veuillez consulter Satisfiability Modulo Theories.

En ce qui concerne les analyseurs d’accès externe, l’analyseur d’accès IAM n’examine pas les journaux d’accès pour déterminer si une entité externe a accédé à une ressource dans votre zone de confiance. Il génère une recherche lorsqu'une politique basée sur les ressources autorise l'accès à une ressource, même si celle-ci n'a pas été accédé par l'entité externe. l’IAM Access Analyzer ne tient pas non plus compte de l'état des comptes externes lorsqu'il réalise sa détermination. En d’autres termes, s’il indique que le compte 111122223333 peut accéder à votre compartiment Amazon S3, il ne sait rien de l’état des utilisateurs, des rôles, des politiques de contrôle des services (SCP) et d’autres configurations pertinentes dans ce compte. Ceci est lié à la confidentialité des clients - l’IAM Access Analyzer ne tient pas compte du propriétaire de l'autre compte. C'est également lié à la sécurité - si le compte n'appartient pas au client de l’IAM Access Analyzer, il est toujours important de savoir qu'une entité externe pourrait accéder à ses ressources même s'il n'y a actuellement aucun principal dans le compte qui pourrait y accéder

l’IAM Access Analyzer ne prend en compte que certaines clés de condition IAM sur lesquelles les utilisateurs externes n'ont pas d'influence directe ou qui ont un impact sur l'autorisation. Pour obtenir des exemples de clés que l’IAM Access Analyzer prend en compte veuillez consulter IAM Access Analyzer filter keys (Clés de filtre Access Analyzer).

À l'heure actuelle, IAM Access Analyzer ne communique pas les résultats des responsables de AWS service ou des comptes de service internes. Dans les rares cas où l'IAM Access Analyzer n'est pas en mesure de déterminer complètement si une instruction de politique octroie l'accès à une entité externe, il se trompe en déclarant un faux résultat positif. l'IAM Access Analyzer est conçu pour fournir une vue exhaustive du partage de ressources dans votre compte et s'efforce de réduire les faux négatifs.

Accès non utilisé

Vous devez créer un analyseur pour les résultats des accès non utilisés pour vos rôles, même si vous avez déjà créé un analyseur pour générer des résultats des accès externes pour vos ressources. Après avoir créé l'analyseur, l'analyseur d'accès IAM examine l'activité d'accès pour identifier les accès non utilisés. IAM Access Analyzer examine les dernières informations consultées pour tous les rôles, clés d'accès utilisateur et mots de passe utilisateur de votre AWS organisation et de vos comptes afin de vous aider à identifier les accès non utilisés. Pour les rôles et utilisateurs IAM actifs, l'analyseur d'accès IAM utilise les informations du dernier accès au service et à l'action IAM pour identifier les autorisations non utilisées. Vous pouvez utiliser des analyseurs d'accès inutilisés pour adapter votre processus de révision au niveau de l' AWS organisation et du compte. Vous pouvez utiliser les dernières informations auxquelles vous avez accédé pour approfondir les rôles individuels.

Tableau de bord récapitulatif

Pour les accès externes et non utilisés, l'analyseur d'accès IAM organise les résultats dans un tableau de bord récapitulatif. Pour l'accès externe, le tableau de bord récapitulatif met en évidence la distinction entre les résultats relatifs à l'accès public et ceux relatifs à l'accès intercompte, et fournit une ventilation des résultats par type de ressource. Pour les accès non utilisés, le tableau de bord met en évidence ceux Comptes AWS qui ont le plus de résultats et fournit une ventilation des résultats par type. Une fois que vous avez créé un analyseur pour les accès externes ou non utilisés, l'analyseur d'accès IAM ajoute automatiquement de nouveaux résultats au tableau de bord axées sur les rôles pour lesquels les autorisations ne sont pas utilisées.

Démarrage avec AWS Identity and Access Management Access Analyzer résultats

Utilisez les informations de cette rubrique pour connaître les exigences nécessaires à l'utilisation et à la gestion AWS Identity and Access Management Access Analyzer, puis comment activer IAM Access Analyzer. Pour en savoir plus sur le rôle lié au service pour l'IAM Access Analyzer, veuillez

consulter Utilisation des rôles liés aux services pour AWS Identity and Access Management Access Analyzer.

Autorisations requises pour l’utilisation de l’IAM Access Analyzer

Pour configurer et utiliser correctement l’IAM Access Analyzer, le compte que vous utilisez doit disposer des autorisations requises.

AWS politiques gérées pour IAM Access Analyzer

AWS Identity and Access Management Access Analyzer fournit des politiques AWS gérées pour vous aider à démarrer rapidement.

- IAM AccessAnalyzer FullAccess - Permet un accès complet à IAM Access Analyzer pour les administrateurs. Cette politique permet également de créer les rôles liés aux services nécessaires pour permettre à IAM Access Analyzer d'analyser les ressources de votre compte ou de votre organisation. AWS
- Accès IAM : autorise AccessAnalyzer ReadOnly l'accès en lecture seule à IAM Access Analyzer. Vous devez ajouter des politiques supplémentaires à vos identités IAM (utilisateurs, groupes d'utilisateurs ou rôles) pour les autoriser à afficher leurs résultats.

Ressources définies par l'analyseur d'accès IAM

Pour afficher les ressources définies par l’analyseur d’accès IAM, consultez Types de ressources définis par l’analyseur d’accès IAM dans la Référence de l’autorisation de service.

Autorisations de service de l’IAM Access Analyzer requises

L’analyseur d’accès IAM utilise un rôle lié au service (SLR) IAM nommé `AWSServiceRoleForAccessAnalyzer`. Ce SLR accorde au service un accès en lecture seule pour analyser les ressources à l'aide de politiques basées sur les AWS ressources et analyser les accès non utilisés en votre nom. Le service crée le rôle dans votre compte dans les scénarios suivants :

- Vous créez un analyseur d’accès externe avec votre compte comme zone de confiance.
- Vous créez un analyseur d’accès non utilisé avec votre compte comme compte sélectionné.

Pour plus d’informations, consultez Utilisation des rôles liés aux services pour AWS Identity and Access Management Access Analyzer.

> **Note**
>
> L'IAM Access Analyzer est régional. Pour les accès externes, vous devez activer l'analyseur d'accès IAM dans chaque région de manière indépendante.
> En cas d'accès non utilisé, les résultats de l'analyseur ne changent pas en fonction de la région. Il n'est pas nécessaire de créer un analyseur dans chaque région où vous disposez de ressources.

Dans certains cas, une fois que vous avez créé un analyseur d'accès externe ou non utilisé dans l'analyseur d'accès IAM, la page ou le tableau de bord des Résultats se charge sans résultats ni récapitulatif. Cela peut être dû à un retard d'affichage de vos résultats dans la console. Vous devrez peut-être actualiser manuellement le navigateur ou y revenir ultérieurement pour afficher vos résultats ou votre récapitulatif. Si vous ne voyez toujours pas de résultats pour un analyseur d'accès externe, c'est parce que vous n'avez pas de ressources prises en charge dans votre compte auxquelles une entité externe peut accéder. Si une politique qui accorde l'accès à une entité externe est appliquée à une ressource, l'IAM Access Analyzer génère un résultat.

> **Note**
>
> Pour les analyseurs d'accès externe, il peut s'écouler jusqu'à 30 minutes après la modification d'une politique pour que l'analyseur d'accès IAM analyse la ressource, puis génère un nouveau résultat d'accès externe ou met à jour un résultat existant pour l'accès à la ressource. Qu'il s'agisse d'analyseurs d'accès externe ou non utilisé, les mises à jour des résultats peuvent ne pas apparaître immédiatement dans le tableau de bord.

Autorisations de l'analyseur d'accès IAM requises pour consulter le tableau de bord des résultats

Pour afficher le Tableau de bord des résultats de l'analyseur d'accès IAM, le compte que vous utilisez doit disposer d'un accès pour effectuer les actions requises suivantes :

- `GetAnalyzer`
- `ListAnalyzers`
- `GetFindingsStatistics`

Pour afficher toutes les actions définies par l'analyseur d'accès IAM, consultez Actions définies par l'analyseur d'accès IAM dans la Référence de l'autorisation de service.

Activation de l'IAM Access Analyzer

Pour créer un analyseur d'accès externe avec Compte AWS la zone de confiance

Pour activer un analyseur d'accès externe dans une région, vous devez créer un analyseur dans cette région. Vous devez créer un analyseur d'accès externe dans chaque région dans laquelle vous souhaitez surveiller l'accès à vos ressources.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Access Analyzer (Analyseur d'accès).
3. Choisissez Paramètres de l'analyseur.
4. Sélectionnez Create Analyzer (créer un analyseur).
5. Dans la section Analyse, sélectionnez Analyse d'accès externes.
6. Dans la section Informations sur l'analyseur, vérifiez que la région affichée est celle dans laquelle vous souhaitez activer l'analyseur d'accès IAM.
7. Entrez un nom pour l'analyseur.
8. Choisissez Courant Compte AWS comme zone de confiance pour l'analyseur.

> **Note**
>
> Si votre compte n'est pas le compte AWS Organizations de gestion ou le compte administrateur délégué, vous ne pouvez créer qu'un seul analyseur avec votre compte comme zone de confiance.

9. Facultatif. Ajoutez les balises que vous souhaitez appliquer à l'analyseur.
10. Sélectionnez Envoyer.

Lorsque vous créez un analyseur d'accès externe pour activer l'analyseur d'accès IAM, un rôle lié au service nommé `AWSServiceRoleForAccessAnalyzer` est créé dans votre compte.

Pour créer un analyseur d'accès externe avec l'organisation comme zone de confiance

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Access Analyzer (Analyseur d'accès).

3. Choisissez Paramètres de l’analyseur.
4. Sélectionnez Create Analyzer (créer un analyseur).
5. Dans la section Analyse, sélectionnez Analyse d’accès externes.
6. Dans la section Informations sur l’analyseur, vérifiez que la région affichée est celle dans laquelle vous souhaitez activer l’analyseur d’accès IAM.
7. Entrez un nom pour l'analyseur.
8. Choisissez Organisation courante comme zone de confiance pour l’analyseur.
9. Facultatif. Ajoutez les balises que vous souhaitez appliquer à l'analyseur.
10. Sélectionnez Envoyer.

Lorsque vous créez un analyseur d’accès externe avec l’organisation comme zone de confiance, un rôle lié au service nommé `AWSServiceRoleForAccessAnalyzer` est créé dans chaque compte de votre organisation.

Pour créer un analyseur d’accès non utilisé pour le compte courant

Suivez la procédure suivante pour créer un analyseur d’accès non utilisé pour un seul Compte AWS. En cas d’accès non utilisé, les résultats de l’analyseur ne changent pas en fonction de la région. Il n’est pas nécessaire de créer un analyseur dans chaque région où vous disposez de ressources.

L’analyseur d’accès IAM facture l’analyse des accès non utilisés en fonction du nombre de rôles et d’utilisateurs IAM analysés par mois et par analyseur. Pour plus d’informations sur les tarifs, consultez la Tarification de l’analyseur d’accès IAM.

1. Ouvrez la console IAM à l’adresse https://console.aws.amazon.com/iam/.
2. Choisissez Access Analyzer (Analyseur d'accès).
3. Choisissez Paramètres de l’analyseur.
4. Sélectionnez Create Analyzer (créer un analyseur).
5. Dans la section Analyse, sélectionnez Analyse des accès non utilisés.
6. Entrez un nom pour l'analyseur.
7. Pour Période de suivi, saisissez le nombre de jours pendant lesquels vous souhaitez générer des résultats concernant les autorisations non utilisées. Par exemple, si vous saisissez 90 jours, l’analyseur générera des résultats pour les entités IAM du compte sélectionné pour toutes les autorisations qui n’ont pas été utilisées depuis 90 jours ou plus depuis la dernière analyse de l’analyseur. Vous pouvez choisir une valeur comprise entre 1 et 180 jours.

8. Pour Comptes sélectionnés, choisissez Courant Compte AWS.

> **Note**
>
> Si votre compte n'est pas le compte AWS Organizations de gestion ou le compte administrateur délégué, vous ne pouvez créer qu'un seul analyseur avec votre compte comme compte sélectionné.

9. Facultatif. Ajoutez les balises que vous souhaitez appliquer à l'analyseur.
10. Sélectionnez Envoyer.

Lorsque vous créez un analyseur d'accès non utilisé pour activer l'analyseur d'accès IAM, un rôle lié au service nommé `AWSServiceRoleForAccessAnalyzer` est créé dans votre compte.

Pour créer un analyseur d'accès non utilisé avec l'organisation actuelle

Utilisez la procédure suivante pour créer un analyseur d'accès inutilisé permettant à une organisation de passer Comptes AWS en revue de manière centralisée l'ensemble de l'organisation. Pour l'analyse des accès non utilisés, les résultats de l'analyseur ne changent pas en fonction de la région. Il n'est pas nécessaire de créer un analyseur dans chaque région où vous disposez de ressources.

L'analyseur d'accès IAM facture l'analyse des accès non utilisés en fonction du nombre de rôles et d'utilisateurs IAM analysés par mois et par analyseur. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

> **Note**
>
> Si le compte d'un membre est supprimé de l'organisation, l'analyseur d'accès non utilisé cessera de générer de nouveaux résultats et de mettre à jour les résultats existants pour ce compte au bout de 24 heures. Les résultats associés au compte membre supprimé de l'organisation seront définitivement supprimés après 90 jours.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Access Analyzer (Analyseur d'accès).
3. Choisissez Paramètres de l'analyseur.
4. Sélectionnez Create Analyzer (créer un analyseur).

5. Dans la section Analyse, sélectionnez Analyse des accès non utilisés.
6. Entrez un nom pour l'analyseur.
7. Pour Période de suivi, saisissez le nombre de jours pendant lesquels vous souhaitez générer des résultats concernant les autorisations non utilisées. Par exemple, si vous saisissez 90 jours, l'analyseur générera des résultats pour les entités IAM dans les comptes de l'organisation sélectionnée pour toutes les autorisations qui n'ont pas été utilisées depuis 90 jours ou plus depuis le dernière analyse de l'analyseur. Vous pouvez choisir une valeur comprise entre 1 et 180 jours.
8. Pour Comptes sélectionnés, choisissez Organisation actuelle comme comptes sélectionnés pour l'analyseur.
9. Facultatif. Ajoutez les balises que vous souhaitez appliquer à l'analyseur.
10. Sélectionnez Envoyer.

Lorsque vous créez un analyseur d'accès non utilisé pour activer l'analyseur d'accès IAM, un rôle lié au service nommé `AWSServiceRoleForAccessAnalyzer` est créé dans votre compte.

Statut de l'IAM Access Analyzer

Pour afficher l'état de vos analyseurs, sélectionnez Analyzers (Analyseurs). Les analyseurs créés pour une organisation ou un compte peuvent avoir l'état suivant :

État	Description
Actif	Pour les analyseurs d'accès externe, l'analyseur surveille activement les ressources dans sa zone de confiance. L'analyseur génère activement de nouveaux résultats et met à jour les résultats existants. Pour les analyseurs d'accès non utilisés, l'analyseur surveille activement les accès non utilisés au sein de l'organisation sélectionnée ou pendant la Compte AWS période de suivi spécifiée. L'analyseur génère activement de nouveaux résultats et met à jour les résultats existants.

État	Description
Création	La création de l'analyseur est toujours en cours. L'analyseur deviendra actif une fois la création terminée.
Désactivées	L'analyseur est désactivé suite à une action de l' AWS Organizations administrateur. Par exemple, le compte de l'analyseur en tant qu'administrateur délégué pour IAM Access Analyzer a pu être supprimé. Lorsque l'analyseur est désactivé, il ne génère pas de nouveaux résultats ou ne met pas à jour les résultats existants.
Échec	La création de l'analyseur a échoué en raison d'un problème de configuration. L'analyseur ne génère pas de résultats. Supprimez l'analyseur et créez-en un nouveau.

Affichage du tableau de bord des résultats de l'analyseur d'accès IAM

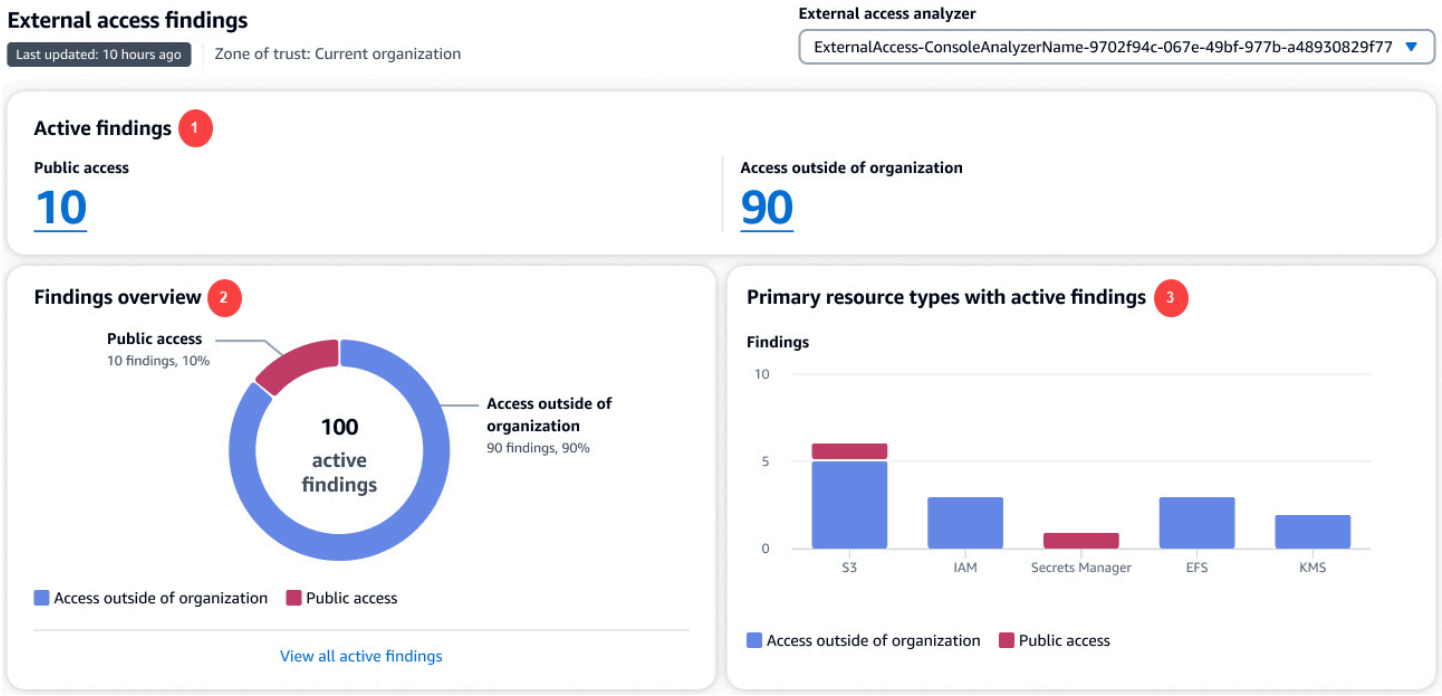
AWS Identity and Access Management Access Analyzer organise les résultats relatifs aux accès externes et aux accès non utilisés dans un tableau de bord récapitulatif visuel. Le tableau de bord vous permet de mieux comprendre l'utilisation efficace des autorisations à grande échelle et d'identifier les comptes qui nécessitent une attention particulière. Vous pouvez utiliser le tableau de bord pour consulter les résultats par AWS organisation, compte et type de recherche.

Pour consulter le tableau de bord récapitulatif des analyseurs d'accès externe

> **Note**
>
> Après avoir créé ou mis à jour un analyseur, le tableau de bord récapitulatif peut mettre du temps à refléter les mises à jour des résultats.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.

2. Choisissez Access Analyzer (Analyseur d'accès). La fenêtre Récapitulatif s'affiche.
3. Choisissez un analyseur dans le menu déroulant Analyseur d'accès externe. Un récapitulatif des résultats de l'analyseur est affiché dans la section Résultats relatifs à l'accès externe.



Dans l'image précédente, le tableau de bord des résultats des accès externes est visible sur la page Récapitulatif :

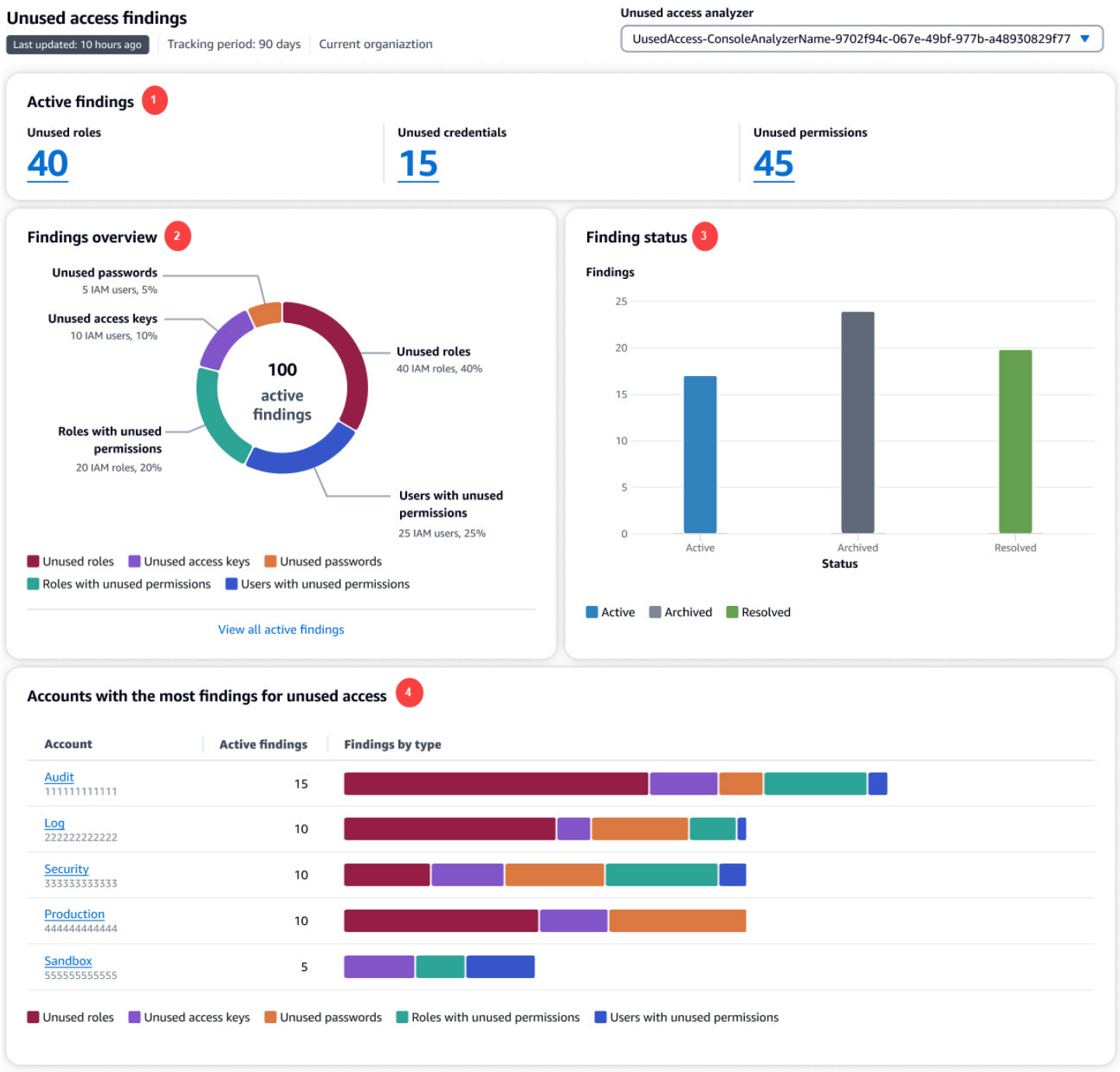
1. La section Résultats actifs inclut le nombre de résultats actifs accessibles au public et le nombre de résultats actifs qui fournissent un accès en dehors du compte ou de l'organisation. Choisissez un nombre pour énumérer tous les résultats actifs de chaque type.
2. La section Aperçu des résultats comprend une ventilation des types de résultats actifs. Choisissez Afficher tous les résultats actifs pour obtenir une liste complète des résultats actifs pour le compte ou l'organisation de l'analyseur.
3. La section Principaux types de ressources présentant des résultats actifs comprend une ventilation des principaux types de ressources présentant des résultats actifs. Ces informations vous aident à hiérarchiser les résultats pour les ressources principales en premier. Par exemple, Amazon S3, DynamoDB et. AWS KMS Cette liste de tous les types de ressources n'est pas exhaustive. Votre analyseur peut contenir des résultats actifs pour des types de ressources non répertoriés dans cette section.

Pour consulter le tableau de bord récapitulatif des analyseurs d’accès non utilisé

L’analyseur d’accès IAM facture l’analyse d’accès non utilisé en fonction du nombre de rôles et d’utilisateurs IAM analysés par mois. Pour plus d’informations sur les tarifs, consultez la Tarification de l’analyseur d’accès IAM.

> **Note**
>
> Après avoir créé ou mis à jour un analyseur, en fonction du nombre d’utilisateurs et de rôles, le tableau de bord récapitulatif peut mettre du temps à refléter les mises à jour des résultats.

1. Ouvrez la console IAM à l’adresse https://console.aws.amazon.com/iam/.
2. Choisissez Access Analyzer (Analyseur d'accès). La fenêtre Récapitulatif s’affiche.
3. Choisissez un analyseur dans le menu déroulant Analyseur d’accès non utilisé. Un récapitulatif des résultats de l’analyseur est affiché dans la section Résultats des accès non utilisés.



Dans l'image précédente, le tableau de bord des résultats des accès externes est visible sur la page Récapitulatif :

1. La section Résultats actifs inclue le nombre de résultats actifs concernant des rôles, des informations d'identification et des autorisations non utilisés dans votre compte ou votre organisation. Les informations d'identification non utilisées incluent à la fois les clés d'accès et les résultats de mots de passe non utilisés. Les autorisations non utilisées incluent à la fois les

utilisateurs et les rôles dont les autorisations ne sont pas utilisées. Choisissez un nombre pour énumérer tous les résultats actifs de chaque type.

2. La section Aperçu des résultats comprend une ventilation des types de résultats actifs. Choisissez Afficher tous les résultats actifs pour obtenir une liste complète des résultats actifs pour le compte ou l'organisation de l'analyseur.
3. La section Statut du résultat inclut une ventilation du statut des résultats (Actif, Archivé et Résolu) pour votre compte ou votre organisation.
4. La section Comptes présentant le plus grand nombre de résultats pour les accès non utilisés ne s'affiche que si les comptes sélectionnés dans votre analyseur d'accès non utilisés se situent au niveau de l'organisation. Il inclut une ventilation des comptes de votre organisation ayant enregistré les résultats les plus actifs. Cette liste n'est pas exhaustive de tous les comptes de votre organisation. Votre analyseur peut contenir des résultats actifs pour d'autres comptes non répertoriés dans cette section.

Utilisation des résultats

Résultats des accès externes

Les résultats d'accès externe sont générés une seule fois pour chaque instance d'une ressource qui est partagée en dehors de votre zone de confiance. Chaque fois qu'une politique basée sur les ressources est modifiée, IAM Access Analyzer analyse la politique. Si la politique mise à jour partage une ressource déjà identifiée dans un résultat, mais avec des autorisations ou des conditions différentes, un nouveau résultat est généré pour cette instance du partage de ressource. Si l'accès dans le premier résultat est supprimé, celui-ci est mis à jour et prend le statut Résolu.

Le statut de tous les résultats reste Actif jusqu'à ce que vous les archiviez ou que vous supprimiez l'accès ayant généré le résultat. Lorsque vous supprimez l'accès, le statut du résultat est mis à jour et devient Résolu.

> **Note**
>
> Lorsque la politique est modifiée, l'analyseur d'accès IAM peut prendre jusqu'à 30 minutes pour analyser la ressource et mettre à jour le résultat d'accès externe.

Résultats des accès non utilisés

Les résultats des accès non utilisés sont générés pour les entités IAM au sein du compte ou de l'organisation sélectionné en fonction du nombre de jours spécifié lors de la création de l'analyseur. Un nouveau résultat est généré la prochaine fois que l'analyseur analyse les entités si l'une des conditions suivantes est remplie :

- Un rôle est inactif pendant le nombre de jours spécifié.
- Une autorisation, un mot de passe utilisateur ou une clé d'accès utilisateur non utilisés dépasse le nombre de jours spécifié.

Vous devez examiner tous les résultats de votre compte pour déterminer si l'accès externe ou non utilisé est attendu et approuvé. Si l'accès externe ou non utilisé identifié dans le résultat est attendu, vous pouvez archiver le résultat. Lorsque vous archivez un résultat, le statut devient Archivé et le résultat est supprimé de la liste des résultats actifs. Le résultat n'est pas supprimé. Vous pouvez consulter vos résultats archivés à tout moment. Passez en revue tous les résultats de votre compte jusqu'à ce qu'il ne reste plus aucun résultat actif. Une fois que vous êtes à zéro résultat, vous savez que tous les nouveaux résultats actifs générés proviennent d'une modification récente de votre environnement.

> **Note**
>
> Les résultats d'accès non utilisés ne sont disponibles qu'à l'aide de l'action API ListFindingsV2.

Examen des résultats

Après l'activation d'IAM Access Analyzer, l'étape suivante est l' examen de tous les résultats afin de déterminer si l'accès identifié dans le résultat est intentionnel ou non. Vous pouvez également examiner les résultats pour déterminer les résultats simillaires pour un accès prévu, puis créer une règle d'archivage pour automatiquement archiver ces résultats. Vous pouvez également examiner les résultats archivés et résolus.

Pour examiner les résultats

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Access Analyzer (Analyseur d'accès).

3. Le tableau de bord des résultats s'affiche. Sélectionnez les résultats actifs pour votre analyseur d'accès externe ou non utilisé.

   Pour de plus d'informations sur l'affichage du tableau de bord des résultats, consultez Affichage du tableau de bord des résultats de l'analyseur d'accès IAM.

> **Note**
>
> Les résultats sont affichés uniquement si vous avez l'autorisation d'afficher les résultats de l'analyseur.

Tous les résultats sont affichés pour l'analyseur. Pour afficher les autres résultats générés par l'analyseur, sélectionnez le type de résultat approprié à partir du menu déroulant Statut :

- Choisissez Active (Actif) pour afficher tous les résultats actifs générés par l'analyseur.
- Choisissez Archived (Archivé) pour afficher uniquement les résultats générés par l'analyseur et archivés. Pour en savoir plus, veuillez consulter la section Archivage des résultats.
- Choisissez Resolved (Résolu) pour afficher uniquement les résultats générés par l'analyseur et résolus. Lorsque vous résolvez le problème qui a généré le résultat, le statut du résultat devient Résolu.

  > **Important**
  >
  > Les résultats résolus sont supprimés 90 jours après la dernière mise à jour du résultat. Les résultats actifs et archivés ne sont pas supprimés, sauf si vous supprimez l'analyseur qui les a générés.

- Choisissez All (Tout) pour afficher tous les résultats dont le statut a été généré par l'analyseur.

Résultats des accès externes

Choisissez Accès externe, puis choisissez l'analyseur d'accès externe dans le menu déroulant Afficher l'analyseur. La page Résultats pour les analyseurs d'accès externe affiche les informations suivantes sur la ressource partagée et la déclaration de politique ayant généré le résultat :

ID de résultat

ID unique affecté au résultat. Choisissez l'ID de résultat pour afficher des détails supplémentaires sur la ressource et l'instruction de politique ayant généré le résultat.

Ressource

Type et nom partiel de la ressource pour laquelle une politique a été appliquée et qui accorde l'accès à une entité externe ne faisant pas partie de votre zone de confiance.

Compte du propriétaire de la ressource

Cette colonne s'affiche uniquement si vous utilisez une organisation comme zone de confiance. Compte de l'organisation propriétaire de la ressource déclarée dans le résultat.

External principal (Principal externe

Principal, qui ne fait pas partie de votre zone de confiance, auquel la politique analysée accorde l'accès. Les valeurs valides sont les suivantes :

- Compte AWS : tous les principaux du Compte AWS répertorié disposant d'autorisations de l'administrateur de ce compte peuvent accéder à la ressource.
- Tout directeur : tous les principaux de tous ceux Compte AWS qui répondent aux conditions incluses dans la colonne Conditions sont autorisés à accéder à la ressource. Par exemple, si un VPC est répertorié, cela signifie que tout principal de n'importe quel compte ayant l'autorisation d'accéder au VPC répertorié peut accéder à la ressource.
- Utilisateur canonique : tous les principaux dont l' Compte AWS ID utilisateur canonique est indiqué sont autorisés à accéder à la ressource.
- IAM role (Rôle IAM) : le rôle IAM répertorié a l'autorisation d'accéder à la ressource.
- IAM user : l'utilisateur IAM répertorié a l'autorisation d'accéder à la ressource.

Condition

Condition de l'instruction de politique qui accorde l'accès. Par exemple, si le champ Condition inclut Source VPC (VPC source), cela signifie que la ressource est partagée avec un principal ayant accès au VPC répertorié. Les conditions peuvent être globales ou spécifiques au service. Les clés de condition globale ont le préfixe `aws:`.

Shared through (Partagé via)

Le champ Shared through (Partagé via) indique comment est accordé l'accès qui a généré le résultat. Les valeurs valides sont les suivantes :

- Politique de compartiment : politique de compartiment attachée au compartiment Amazon S3.
- Access control list (Liste de contrôle d'accès) : liste de contrôle d'accès (ACL) attachée au compartiment Amazon S3.
- Point d'accès : point d'accès ou point d'accès multi-région associé au compartiment Amazon S3. L'ARN du point d'accès est affiché dans les détails des résultats.

Niveau d'accès

Niveau d'accès accordé à l'entité externe par les actions de la politique basée sur les ressources. Consultez les détails du résultat pour plus d'informations. Les valeurs de niveau d'accès sont les suivantes :

- List (Liste) : autorisation de répertorier les ressources au sein du service afin de déterminer si un objet existe. Les actions associées à ce niveau d'accès peuvent répertorier les objets mais ne peuvent pas voir le contenu d'une ressource.
- Read (Lecture) : autorisation de lire (sans modifier) le contenu et les attributs des ressources du service.
- Write (Écriture) : autorisation de créer, supprimer ou modifier les ressources du service.
- Permissions (Autorisations) : autorisation d'octroyer ou de modifier des autorisations sur des ressources dans le service.
- Tagging (Balisage) : autorisations d'effectuer des actions qui modifient uniquement l'état des balises de ressource.

Mis à jour

Horodatage de la mise à jour la plus récente du statut de résultat, ou heure et date auxquelles le résultat a été généré si aucune mise à jour n'a été effectuée.

> **Note**
>
> Lorsqu'une politique est modifiée, IAM Access Analyzer peut mettre jusqu'à 30 minutes afin d’ analyser à nouveau la ressource ainsi que mettre à jour le résultat.

Statut

Statut du résultat : Active (Actif), Archived (Archivé) ou Resolved (Résolu).

Résultats des accès non utilisés

L'analyseur d'accès IAM facture l'analyse d'accès non utilisé en fonction du nombre de rôles et d'utilisateurs IAM analysés par mois. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

Choisissez Accès non utilisé, puis choisissez l'analyseur d'accès non utilisé dans le menu déroulant Afficher l'analyseur. La page Résultats des analyseurs d'accès non utilisés affiche les informations suivants sur l'entité IAM qui a généré le résultat :

ID de résultat

ID unique affecté au résultat. Choisissez l'ID du résultat pour afficher des détails supplémentaires sur l'entité IAM qui a généré le résultat.

Type de résultat

Type de résultat d'accès non utilisé : Clé d'accès non utilisée, Mot de passe non utilisé, Autorisation non utilisée, ou Rôle non utilisé.

Entité IAM

L'entité IAM mentionnée dans le résultat. Il peut s'agir d'un utilisateur ou d'un rôle IAM.

Compte AWS ID

Cette colonne s'affiche uniquement si vous configurez l'analyseur pour tous les membres de Comptes AWS l'organisation. Compte AWS Dans l'organisation propriétaire de l'entité IAM indiquée dans le résultat.

Dernière mise à jour

La dernière fois que l'entité IAM indiquée dans le résultat a été mise à jour, ou la date de création de l'entité si aucune mise à jour n'a été effectuée.

Statut

Statut du résultat : (Actif, Archivé ou Résolu).

Filtrage des résultats

Le filtrage par défaut de la page de résultat consiste à afficher tous les résultats. Pour afficher les résultats actifs, choisissez le statut Actif dans le menu déroulant Statut. Pour consulter les résultats

archivés, choisissez le statut Archivé dans le menu déroulant Statut. Lorsque vous commencez à utiliser IAM Access Analyzer, il n'y a aucun résultat archivé.

Utilisez des filtres pour afficher uniquement les résultats qui répondent aux critères de propriété spécifiés. Pour créer un filtre, sélectionnez la propriété à filtrer, puis choisissez si la propriété est égale ou contient une valeur, puis saisissez ou choisissez une valeur de propriété à filtrer. Par exemple, pour créer un filtre qui affiche uniquement les résultats relatifs à une propriété spécifique Compte AWS, choisissez AWS Compte pour la propriété, puis choisissez AWS Compte =, puis entrez le numéro de compte de la propriété pour Compte AWS laquelle vous souhaitez consulter les résultats.

Pour obtenir la liste des clés de filtre que vous pouvez utiliser pour créer ou mettre à jour une règle d'archivage, veuillez consulter Clés de filtre de l’IAM Access Analyzer.

Filtrer les résultats des accès externes

Pour filtrer les résultats des accès externes

1. Choisissez Accès externe, puis choisissez l’analyseur dans la liste déroulante Afficher l’analyseur.
2. Choisissez la case de recherche pour afficher une liste des propriétés disponibles.
3. Choisissez la propriété à utiliser pour filtrer les résultats affichés.
4. Choisissez la valeur à faire correspondre pour la propriété. Seuls les résultats ayant cette valeur dans le résultat sont affichés.

   Par exemple, choisissez Ressource comme propriété, puis Ressource :, puis saisissez tout ou partie du nom d’un compartiment et appuyez sur Entrée. Seuls les résultats pour le compartiment qui correspond aux critères du filtre sont affichés. Pour créer un filtre qui affiche uniquement les résultats pour les ressources autorisant l’accès public, vous pouvez choisir la propriété Accès public, puis Accès public =, puis Accès public = true.

Vous pouvez ajouter des propriétés supplémentaires pour filtrer davantage les résultats affichés. Lorsque vous ajoutez des propriétés supplémentaires, seuls les résultats correspondant à toutes les conditions du filtre sont affichés. La définition d'un filtre pour afficher les résultats correspondant à une propriété OU à une autre propriété n'est pas prise en charge. Choisissez Effacer les filtres pour effacer tous les filtres que vous avez définis et afficher tous les résultats avec le statut spécifié pour votre analyseur.

Certains champs ne s'affichent que lorsque vous veuillez consulter les résultats d'un analyseur dont la zone de confiance est une organisation.

Les propriétés suivantes sont disponibles pour la définition de filtres :

- Public access (Accès public) : pour filtrer sur les résultats des ressources autorisant l'accès public, filtrez par Public access (Accès public), puis Public access: true (Accès public : vrai).
- Resource (Ressource) : pour filtrer par ressource, saisissez tout ou partie du nom de la ressource.
- Resource Type (Type de ressource) : pour filtrer par type de ressource, sélectionnez le type dans la liste affichée.
- Compte propriétaire de la ressource : utilisez cette propriété pour filtrer selon le compte de l'organisation propriétaire de la ressource indiquée dans le résultat.
- AWS Compte — Utilisez cette propriété pour filtrer en fonction Compte AWS des accès autorisés dans la section Principal d'une déclaration de politique. Pour filtrer par Compte AWS, saisissez tout ou partie de l' Compte AWS identifiant à 12 chiffres, ou tout ou partie de l'ARN complet du compte de l' AWS utilisateur ou du rôle externe ayant accès aux ressources du compte courant.
- Utilisateur canonique : pour filtrer par utilisateur canonique, saisissez l'ID d'utilisateur canonique comme défini pour les compartiments Amazon S3. Pour en savoir plus, veuillez consulter Identificants de compte AWS.
- Federated User (Utilisateur fédéré) : pour filtrer par utilisateur fédéré, saisissez tout ou partie de l'ARN de l'identité fédérée. Pour en savoir plus, veuillez consulter Fournisseurs d'identité et fédération.
- ID de résultat : pour filtrer par ID de résultat, saisissez tout ou une partie de l'ID de résultat.
- ARN principal — Utilisez cette propriété pour filtrer l'ARN du principal (utilisateur, rôle ou groupe IAM) utilisé dans une clé de PrincipalArn condition aws :. Pour filtrer par ARN principal, saisissez tout ou partie de l'ARN de l'utilisateur, du rôle ou du groupe IAM provenant d'un utilisateur externe Compte AWS indiqué dans une recherche.
- Principal OrgID (OrgID de principal) : pour filtrer par OrgID de principal, saisissez tout ou partie de l'ID d'organisation associé aux principaux externes qui appartiennent à l'organisation AWS spécifiée comme condition dans le résultat. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- Principal OrgPaths — Pour filtrer par principal OrgPaths, saisissez tout ou partie de l'ID de l' AWS organisation ou de l'unité organisationnelle (UO) qui autorise l'accès à tous les principaux externes membres du compte de l'organisation ou de l'UO spécifiée comme condition dans la politique. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.

- Compte source : pour filtrer sur le compte source, saisissez tout ou partie de l' Compte AWS identifiant associé aux ressources, tel qu'il est utilisé dans certaines autorisations interservices dans AWS. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- Source ARN (ARN source) : pour filtrer par ARN source, saisissez tout ou partie de l'ARN spécifié comme condition dans le résultat. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- Source IP (IP source) : pour filtrer par IP source, saisissez tout ou partie de l'adresse IP qui permet aux entités externes d'accéder aux ressources du compte actuel lors de l'utilisation de l'adresse IP spécifiée. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- Source VPC (VPC source) : pour filtrer par VPC source, saisissez tout ou partie de l'ID de VPC qui permet aux entités externes d'accéder aux ressources du compte actuel lors de l'utilisation du VPC spécifié. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- OrGid source : pour filtrer par OrgID source, saisissez tout ou partie de l'ID d'organisation associé aux ressources, tel qu'il est utilisé dans certaines autorisations interservices dans. AWS Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- Source OrgPaths : pour filtrer par source OrgPaths, saisissez tout ou partie de l'unité organisationnelle (UO) associée aux ressources, telle qu'elle est utilisée dans certaines autorisations interservices dans AWS. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- ID utilisateur — Pour filtrer par ID utilisateur, saisissez tout ou partie de l'ID utilisateur de l'utilisateur IAM auprès d'un utilisateur Compte AWS externe autorisé à accéder aux ressources du compte courant. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.
- ID de clé KMS — Pour filtrer par ID de clé KMS, saisissez tout ou partie de l'ID de clé KMS spécifié comme condition pour l'accès AWS KMS chiffré à un objet Amazon S3 sur votre compte courant.
- Google Audience (Audience Google) : pour filtrer par audience Google, saisissez tout ou partie de l'ID de l'application Google spécifiée comme condition pour l'accès au rôle IAM dans votre compte actuel. Pour en savoir plus, consultez les rubriques IAM et clés AWS STS contextuelles de condition.
- Audience Cognito : pour filtrer par audience Amazon Cognito, saisissez tout ou une partie de l’ID de réserve d’identités Amazon Cognito spécifié comme condition pour l’accès au rôle IAM dans votre compte actuel. Pour en savoir plus, consultez les rubriques IAM et clés AWS STS contextuelles de condition.
- Compte appelant : Compte AWS ID du compte qui possède ou contient l'entité appelante, tel qu'un rôle IAM, un utilisateur ou un utilisateur root du compte. Ceci est utilisé par les services

qui appellent AWS KMS. Pour filtrer par compte d'appelant, saisissez tout ou une partie de l'ID Compte AWS .

- Facebook App ID (ID d'application Facebook) : pour filtrer par ID d'application Facebook, saisissez tout ou partie de l'ID d'application Facebook (ou de l'ID de site) spécifié comme condition pour autoriser la connexion avec accès de fédération Facebook à un rôle IAM dans votre compte actuel. Pour en savoir plus, veuillez consulter la section ID dans la rubrique Clés de contexte de condition IAM et AWS STS.
- Amazon App ID (ID d'application Amazon) : pour filtrer par ID d'application Amazon, saisissez tout ou partie de l'ID d'application Amazon (ou de l'ID de site) spécifié comme condition pour autoriser l'accès de fédération Login with Amazon à un rôle IAM dans votre compte actuel. Pour en savoir plus, veuillez consulter la section ID dans la rubrique Clés de contexte de condition IAM et AWS STS.
- Lambda Event Source Token (Jeton de source d'événement Lambda) : pour filtrer sur le jeton de source d'événement Lambda transmis avec des intégrations Alexa, saisissez tout ou partie de la chaîne du jeton.

Filtrer les résultats des accès non utilisés

Pour filtrer les résultats des accès non utilisés

1. Choisissez Accès non utilisé, puis choisissez l'analyseur dans la menu déroulant Afficher l'analyseur.
2. Choisissez la case de recherche pour afficher une liste des propriétés disponibles.
3. Choisissez la propriété à utiliser pour filtrer les résultats affichés.
4. Choisissez la valeur à faire correspondre pour la propriété. Seuls les résultats ayant cette valeur dans le résultat sont affichés.

   Par exemple, choisissez Type de résultats comme propriété, puis choisissez Type de résultats =, puis choisissez Type de résultats = UnuseDiamRole. Seuls les résultats avec un type UnuseDiamRole sont affichés.

Vous pouvez ajouter des propriétés supplémentaires pour filtrer davantage les résultats affichés. Lorsque vous ajoutez des propriétés supplémentaires, seuls les résultats correspondant à toutes les conditions du filtre sont affichés. La définition d'un filtre pour afficher les résultats correspondant à une propriété OU à une autre propriété n'est pas prise en charge. Choisissez Effacer les filtres pour

effacer tous les filtres que vous avez définis et afficher tous les résultats avec le statut spécifié pour votre analyseur.

Les champs suivants s'affichent uniquement lorsque vous consultez les résultats d'un analyseur qui surveille les accès non utilisés :

- Type de résultats : pour filtrer par type de recherche, filtrez par type de résultats, puis choisissez le type de recherche.
- Resource (Ressource) : pour filtrer par ressource, saisissez tout ou partie du nom de la ressource.
- Resource Type (Type de ressource) : pour filtrer par type de ressource, sélectionnez le type dans la liste affichée.
- Compte propriétaire de la ressource : utilisez cette propriété pour filtrer selon le compte de l’organisation propriétaire de la ressource indiquée dans le résultat.
- Identifiant de recherche — Pour filtrer par ID de recherche, saisissez tout ou partie de l'ID de recherche.

Archivage des résultats

Lorsque vous obtenez un résultat pour accéder à une ressource intentionnelle, vous pouvez archiver les résultats. Par exemple, un résultat des accès externes pour un rôle IAM utilisé par plusieurs utilisateurs pour des flux de travail approuvés ou un résultat des accès non utilisés pour une clé d’accès qui peut toujours être nécessaire. Lorsque vous archivez un résultat, il est effacé de la liste des résultats actifs. Les résultats archivés ne sont pas supprimés. Vous pouvez filtrer la page Résultats pour afficher vos résultats archivés et les désarchiver à tout moment.

Pour archiver les résultats à partir de la page Findings (Résultats)

1. Sélectionnez la case à cocher en regard d'un ou plusieurs résultats à archiver.
2. Choisissez Actions, puis Archiver.

   Une confirmation s'affiche en haut de l'écran.

Pour archiver les résultats à partir de la page Détails des résultats

1. Choisissez l'ID de résultat du résultat à archiver.
2. Choisissez Archive (Archiver).

Une confirmation s'affiche en haut de l'écran.

Pour désarchiver les résultats, répétez les étapes précédentes, mais sélectionnez Unarchive (Désarchiver) plutôt que Archive (Archiver). Lorsque vous désarchivez un résultat, son statut devient Active (Actif).

Résolution des résultats

Résultats des accès externes

Pour résoudre les résultats des accès externes générés à partir d'un accès que vous n'aviez pas l'intention d'autoriser, modifiez la déclaration de politique pour supprimer les autorisations qui accordent l'accès à la ressource identifiée. Par exemple, pour les résultats sur les compartiments S3, utilisez la console Amazon S3 pour configurer les autorisations sur le compartiment. Pour les rôles IAM, utilisez la console IAM pour modifier la politique d'approbation du rôle IAM répertorié. Utilisez la console pour les autres ressources prises en charge pour modifier les instructions de politique ayant abouti à un résultat généré.

Lorsque vous apportez une modification afin de résoudre un résultat des accès externes, comme la modification d'une politique appliquée à un rôle IAM, l'analyseur d'accès IAM analyse de nouveau la ressource. Si la ressource n'est plus partagée en dehors de votre zone d'approbation, le statut du résultat devient Resolved (Résolu). Le résultat n'est plus affiché dans la liste des résultats actifs, mais plutôt dans la liste des résultats résolus.

> **Note**
>
> Cela ne s'applique pas aux résultats signalant une Erreur. Lorsque l'Analyseur d'accès IAM n'est pas en mesure d'analyser une ressource, il génère un résultat signalant une erreur. Si vous résolvez le problème qui a empêché l'Analyseur d'accès IAM d'analyser la ressource, le résultat d'erreur est complètement supprimé. Son état ne change pas pour indiquer la résolution du problème.

Si les modifications apportées ont entraîné le partage de la ressource en dehors de votre zone d'approbation, mais d'une manière différente, par exemple avec un principal différent ou pour une autorisation différente, IAM Access Analyzer génère un nouveau résultat actif.

> **Note**
>
> Lorsqu'une politique est modifiée, IAM Access Analyzer peut mettre jusqu'à 30 minutes afin d'analyser à nouveau la ressource ainsi que mettre à jour le résultat. Les résultats résolus sont supprimés 90 jours après la dernière mise à jour du résultat.

Résultats des accès non utilisés

Pour les résultats inutilisés de l'analyseur d'accès, IAM Access Analyzer propose des étapes recommandées pour résoudre les résultats en fonction du type de résultat.

Une fois que vous avez apporté une modification pour résoudre un résultat des accès non utilisés, le statut du résultat passe à Résolu lors de la prochaine exécution de l'analyseur d'accès non utilisé. Le résultat n'est plus affiché dans la liste des résultats actifs, mais dans la liste des résultats résolus. Si vous apportez une modification qui ne résoud que partiellement à un résultat des accès non utilisés, le résultat existant devient Résolu mais un nouveau est généré. Par exemple, vous supprimez uniquement certaines autorisations non utilisées dans un résultat, mais pas tous.

L'analyseur d'accès IAM facture l'analyse d'accès non utilisé en fonction du nombre de rôles et d'utilisateurs IAM analysés par mois. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

Résolution des problèmes d'autorisation non utilisés

En cas de détection d'autorisations non utilisées, IAM Access Analyzer peut recommander des politiques à supprimer d'un utilisateur ou d'un rôle IAM et fournir de nouvelles politiques pour remplacer les politiques d'autorisation existantes. La recommandation de stratégie n'est pas prise en charge pour les scénarios suivants :

- La recherche d'autorisation non utilisée concerne un utilisateur IAM appartenant à un groupe d'utilisateurs.
- La recherche d'autorisation non utilisée concerne un rôle IAM pour IAM Identity Center.
- La recherche d'autorisation non utilisée est associée à une politique d'autorisation existante qui inclut l'`notAction`élément.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Accès non utilisé.

3. Choisissez un résultat avec le type de recherche « Autorisations non utilisées ».
4. Dans la section Recommandations, si des politiques sont répertoriées dans la colonne Stratégie recommandée, choisissez Prévisualiser la politique pour afficher la stratégie existante avec la stratégie recommandée pour remplacer la stratégie existante. S'il existe plusieurs politiques recommandées, vous pouvez choisir la stratégie suivante et la stratégie précédente pour afficher chacune des politiques existantes et recommandées.
5. Choisissez Télécharger le JSON pour télécharger un fichier .zip contenant les fichiers JSON de toutes les politiques recommandées.
6. Créez et associez les politiques recommandées à l'utilisateur ou au rôle IAM. Pour plus d'informations, consultez Modifier les autorisations d'un utilisateur (console) et Modifier la politique d'autorisations d'un rôle (console).
7. Supprimez les politiques répertoriées dans la colonne Politique d'autorisation existante de l'utilisateur ou du rôle IAM. Pour plus d'informations, consultez Supprimer une autorisation d'un utilisateur (console) et Modifier une politique d'autorisations de rôle (console).

Résolution des problèmes liés aux rôles non utilisés

En cas de détection de rôles inutilisés, IAM Access Analyzer recommande de supprimer le rôle IAM inutilisé.

1. Ouvrez la console IAM à l’adresse https://console.aws.amazon.com/iam/.
2. Choisissez Accès non utilisé.
3. Choisissez un résultat avec le type de recherche « Rôle non utilisé ».
4. Dans la section Recommandations, passez en revue les détails du rôle IAM.
5. Supprimez le rôle IAM. Pour plus d'informations, consultez Supprimer un rôle IAM (console).

Résolution des problèmes d'accès non utilisés : principaux résultats

En cas de détection d'une clé d'accès non utilisée, IAM Access Analyzer recommande de désactiver ou de supprimer la clé d'accès non utilisée.

1. Ouvrez la console IAM à l’adresse https://console.aws.amazon.com/iam/.
2. Choisissez Accès non utilisé.
3. Choisissez une recherche avec le type de recherche Clés d'accès non utilisées.
4. Dans la section Recommandations, passez en revue les détails de la clé d'accès.

5. Désactivez ou supprimez la clé d'accès. Pour plus d'informations, consultez la section Gestion des clés d'accès (console).

Résolution des problèmes de mots de passe inutilisés

En cas de détection de mots de passe inutilisés, IAM Access Analyzer recommande de supprimer le mot de passe inutilisé pour l'utilisateur IAM.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Accès non utilisé.
3. Choisissez une recherche avec le type de recherche Mot de passe inutilisé.
4. Dans la section Recommandations, passez en revue les informations relatives à l'utilisateur IAM.
5. Supprimez le mot de passe de l'utilisateur IAM. Pour plus d'informations, consultez la section Création, modification ou suppression d'un mot de passe utilisateur IAM (console).

Types de ressources de l'analyseur d'accès IAM pour l'accès externe

Pour les analyseurs d'accès externes, IAM Access Analyzer analyse les politiques basées sur les ressources qui sont appliquées aux AWS ressources de la région dans laquelle vous avez activé IAM Access Analyzer. Seules les politiques basées sur les ressources sont analysées. Consultez les informations relatives à chaque ressource afin d' obtenir des détails sur la façon dont IAM Access Analyzer génère les résultats pour chaque type de ressource.

> **Note**
>
> Les types de ressources pris en charge répertoriés concernent les analyseurs d'accès externe. Les analyseurs d'accès non utilisés ne prennent en charge que les utilisateurs et les rôles IAM. Pour plus d'informations, consultez Utilisation des résultats.

Types de ressources pris en charge pour l'accès externe :

- Compartiments Amazon Simple Storage Service
- Compartiments du répertoire Amazon Simple Storage Service
- AWS Identity and Access Management rôles
- AWS Key Management Service clés

- AWS Lambda fonctions et couches
- Files d'attente Amazon Simple Queue Service
- AWS Secrets Manager secrets
- Rubriques Amazon Simple Notification Service
- Instantanés volumes Amazon Elastic Block Store
- Instantanés de base de données service de base de données relationnelle Amazon
- Instantanés de cluster de base de données service base de données relationnelle Amazon
- Référentiels Amazon Elastic Container Registry
- Systèmes de fichiers Amazon Elastic File System
- Streams Amazon DynamoDB
- Tables Amazon DynamoDB

Compartiments Amazon Simple Storage Service

Lorsque l'IAM Access Analyzer analyse des compartiments Amazon S3, il génère un résultat si une politique de compartiment Amazon S3, une liste ACL ou un point d'accès, y compris un point d'accès multi-région appliquée à un compartiment accorde l'accès à une entité externe. Une entité externe est un principal ou une autre entité que vous pouvez utiliser pour créer un filtre qui ne se trouve pas dans votre zone d'approbation. Par exemple, si une politique de compartiment accorde l'accès à un autre compte ou autorise un accès public, IAM Access Analyzer génère un résultat. Toutefois, si vous activez le blocage de l'accès public sur votre compartiment, vous pouvez bloquer l'accès au niveau du compte ou du compartiment.

> **Note**
>
> IAM Access Analyzer n'analyse pas la politique de point d'accès associée aux points d'accès intercompte, car le point d'accès et sa politique ne font pas partie du compte de l'analyseur. IAM Access Analyzer génère un résultat public lorsqu'un compartiment délègue l'accès à un point d'accès intercompte et que l'option Bloquer l'accès public n'est pas activée sur le compartiment ou le compte. Lorsque vous activez Bloquer l'accès public, le résultat public est résolu et IAM Access Analyzer génère un résultat intercompte pour le point d'accès intercompte.

Les paramètres Amazon S3 de blocage de l'accès public remplacent les politiques de compartiment qui sont appliquées au compartiment. Les paramètres remplacent également les politiques de point d'accès appliquées aux points d'accès du compartiment. Chaque fois qu'une politique change, IAM Access Analyzer analyse les paramètres de blocage de l'accès public au niveau du compartiment. Cependant, il évalue les paramètres de blocage d'accès public au niveau du compte seulement une fois toutes les 6 heures. Cela signifie que l'IAM Access Analyzer peut ne pas générer ou résoudre un résultat pour un accès public à un compartiment pendant une période pouvant aller jusqu'à 6 heures. Par exemple, si vous disposez d'une politique de compartiment qui autorise l'accès public, IAM Access Analyzer génère un résultat pour cet accès. Si vous activez le blocage de l'accès public afin de bloquer tout accès public au compartiment au niveau du compte, IAM Access Analyzer ne résout pas le résultat pour la politique de compartiment pendant une période pouvant aller jusqu'à 6 heures, même si tout l'accès public au compartiment est bloqué. La résolution des résultats publics concernant les points d'accès intercompte peut également prendre jusqu'à 6 heures une fois que vous avez activé l'option Bloquer l'accès public au niveau du compte.

Pour un point d'accès multi-région, IAM Access Analyzer utilise une politique établie dans le but de générer des résultats. IAM Access Analyzer évalue les modifications apportées aux points d'accès multi-région toutes les 6 heures. Cela signifie que l'IAM Access Analyzer ne génère pas ou ne résout pas un résultat pendant au moins 6 heures, même si vous créez ou supprimez un point d'accès multi-région, ou si vous mettez à jour la politique pour ce point.

Compartiments du répertoire Amazon Simple Storage Service

Les compartiments de répertoire Amazon S3 utilisent la classe de stockage Amazon S3 Express One, recommandée pour les charges de travail ou les applications critiques en termes de performances. Pour les compartiments de répertoires Amazon S3, l'analyseur d'accès IAM analyse la politique de compartiments de répertoires, y compris les déclarations de condition dans une politique, qui permettent à une entité externe d'accéder à un compartiment de répertoires. Pour plus d'informations sur les compartiments de répertoire Amazon S3, consultez Compartiments de répertoire dans le guide de l'utilisateur Amazon Simple Storage Service.

AWS Identity and Access Management rôles

Pour les rôles IAM, IAM Access Analyzer analyse les politiques de confiance. Dans une politique d'approbation de rôle, vous définissez les principaux auxquels vous faites confiance pour endosser le rôle. Une politique d'approbation de rôle est une politique basée sur les ressources requise qui est attachée à un rôle dans IAM. IAM Access Analyzer génère des résultats pour les rôles dans la zone de confiance à laquelle peut accéder une entité externe située en dehors de votre zone de confiance.

> **Note**
>
> Un rôle IAM est une ressource globale. Si une politique d'approbation de rôle accorde l'accès à une entité externe, IAM Access Analyzer génère un résultat dans chaque région activée.

AWS Key Management Service clés

En AWS KMS keys effet, IAM Access Analyzer analyse les politiques clés et les autorisations appliquées à une clé. IAM Access Analyzer génère un résultat si une politique de clé ou d' un octroi autorise l' accès d'une entité externe à la clé. Par exemple, si vous utilisez la clé de CallerAccount condition kms : dans une déclaration de politique pour autoriser l'accès à tous les utilisateurs d'un AWS compte spécifique, et que vous spécifiez un compte autre que le compte courant (la zone de confiance de l'analyseur actuel), IAM Access Analyzer génère un résultat. Pour en savoir plus sur les clés de AWS KMS condition dans les déclarations de politique IAM, consultez la section Clés de AWS KMS condition.

Lorsque l'IAM Access Analyzer analyse une clé KMS, il lit les métadonnées de clé, telles que la politique de clé et la liste des autorisations. Si la politique de clé n'autorise pas le rôle IAM Access Analyzer à lire les métadonnées de clé, un résultat d'erreur de type Accès refusé est généré. Par exemple, si l'instruction de politique suivante est la seule politique appliquée à une clé, un résultat d'erreur de type Accès refusé est généré dans IAM Access Analyzer.

```
{
    "Sid": "Allow access for Key Administrators",
    "Effect": "Allow",
    "Principal": {
       "AWS": "arn:aws:iam::111122223333:role/Admin"
    },
    "Action": "kms:*",
    "Resource": "*"
}
```

Comme cette instruction autorise uniquement le rôle nommé Admin du AWS compte 111122223333 à accéder à la clé, un message d'erreur d'accès refusé est généré car IAM Access Analyzer n'est pas en mesure d'analyser complètement la clé. Un résultat d'erreur s'affiche sous la forme d'un texte en rouge dans le tableau Findings (Résultats). Le résultat ressemble à ce qui suit :

```
{
```

```
    "error": "ACCESS_DENIED",
    "id": "12345678-1234-abcd-dcba-111122223333",
    "analyzedAt": "2019-09-16T14:24:33.352Z",
    "resource": "arn:aws:kms:us-
west-2:1234567890:key/1a2b3c4d-5e6f-7a8b-9c0d-1a2b3c4d5e6f7g8a",
    "resourceType": "AWS::KMS::Key",
    "status": "ACTIVE",
    "updatedAt": "2019-09-16T14:24:33.352Z"
}
```

Lorsque vous créez une clé KMS, les autorisations accordées pour accéder à la clé dépendent de la façon dont vous créez celle-ci. Si vous recevez un résultat d'erreur de type Accès refusé pour une ressource de clé, appliquez l'instruction de politique suivante à la ressource de clé pour accorder à IAM Access Analyzer l'autorisation d'accéder à la clé.

```
{
    "Sid": "Allow IAM Access Analyzer access to key metadata",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/aws-service-role/access-
analyzer.amazonaws.com/AWSServiceRoleForAccessAnalyzer"
        },
    "Action": [
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms:List*"
    ],
    "Resource": "*"
},
```

Après la réception d'un résultat d'erreur de type Accès refusé pour une ressource de clé KMS, puis la résolution du résultat par la mise à jour de la politique de clé, le résultat est mis à jour et prend le statut Resolved (Résolu). S'il existe des instructions de politique ou des autorisations de clé qui accordent l'autorisation sur la clé à une entité externe, vous pouvez voir des résultats supplémentaires pour la ressource clé.

AWS Lambda fonctions et couches

Pour les AWS Lambda fonctions, IAM Access Analyzer analyse les politiques, y compris les déclarations de condition figurant dans une politique, qui accordent l'accès à la fonction à une entité externe. Avec Lambda, vous pouvez associer des politiques uniques basées sur les ressources aux

fonctions, aux versions, aux alias et aux couches. IAM Access Analyzer signale les accès externes en fonction des politiques basées sur les ressources associées aux fonctions et aux couches. IAM Access Analyzer ne signale pas les accès externes sur la base de politiques basées sur les ressources associées aux alias et de versions spécifiques invoquées à l'aide d'un ARN qualifié.

Pour plus d'informations, consultez les sections Utilisation de politiques basées sur les ressources pour Lambda et Utilisation de versions dans le Guide du développeur. AWS Lambda

Files d'attente Amazon Simple Queue Service

Pour les files d'attente Amazon SQS, IAM Access Analyzer analyse les politiques, notamment les instructions de condition dans une politique, permettant à une entité externe l’accès à une file d'attente.

AWS Secrets Manager secrets

Pour détecter AWS Secrets Manager les secrets, IAM Access Analyzer analyse les politiques, y compris les déclarations de condition figurant dans une stratégie, qui permettent à une entité externe d'accéder à un secret.

Rubriques Amazon Simple Notification Service

IAM Access Analyzer analyse les politiques basées sur les ressources associées aux rubriques Amazon SNS, y compris les conditions énoncées dans les politiques qui autorisent l'accès externe à une rubrique. Vous pouvez autoriser les comptes externes à effectuer des actions Amazon SNS, telles que l'abonnement à des sujets et la publication de sujets, par le biais d'une politique basée sur les ressources. Une rubrique Amazon SNS est accessible de l'extérieur si les responsables d'un compte situé en dehors de votre zone de confiance peuvent effectuer des opérations sur le sujet. Lorsque vous choisissez `Everyone` dans votre politique lors de la création d'un sujet Amazon SNS, vous le rendez accessible au public. `AddPermission` est une autre façon d'ajouter une politique basée sur les ressources à une rubrique Amazon SNS qui autorise l'accès externe.

Instantanés volumes Amazon Elastic Block Store

Les instantanés volume d'Amazon Elastic Block Store n’ont pas de politique basée sur les ressources. Un instantané est partagé via les autorisations de partage Amazon EBS. Pour les instantanés volume Amazon EBS, IAM Access Analyzer analyse les listes de contrôle d'accès qui permettent à une entité externe d'accéder à un instantané. Un instantané d'un volume Amazon EBS peut être partagé avec des comptes externes lorsqu'il est crypté. Un instantané de volume non chiffré peut être partagé avec des comptes externes et accorder un accès public. Les paramètres de

partage se trouvent dans l'attribut `CreateVolumePermissions` de l'instantané. Lorsque les clients prévisualisent l'accès externe à un instantané Amazon EBS, ils peuvent spécifier la clé de chiffrement pour indiquer que l'instantané est chiffré, de la même manière qu'IAM Access Analyzer Preview gère les secrets de Secrets Manager.

Instantanés de base de données service de base de données relationnelle Amazon

Les Instantanés de base de données Amazon RDS ne disposent pas de politiques basées sur les ressources. Un instantané de base de données est partagé via les autorisations de base de données Amazon RDS, et seuls les instantanés de base de données manuels peuvent être partagés. Pour les instantanés de base de données Amazon RDS, IAM Access Analyzer analyse les listes de contrôle d'accès qui permettent à une entité externe d'accéder à un instantané. Les instantanés de base de données non cryptés peuvent être publics. Les instantanés de base de données chiffrés ne peuvent pas être partagés publiquement, mais ils peuvent être partagés avec jusqu'à 20 autres comptes. Pour de plus amples informations, veuillez consulter Création d'un instantané de base de données. IAM Access Analyzer considère la possibilité d'exporter un instantané manuel de base de données (par exemple, vers un compartiment Amazon S3) comme un accès sécurisé.

> **Note**
>
> IAM Access Analyzer n'identifie pas les accès publics ou l’accès-intercompte configurés directement dans la base de données elle-même. IAM Access Analyzer identifie uniquement les résultats relatifs à l'accès public ou l’accès-intercompte configuré sur l'instantané de base de données Amazon RDS.

Instantanés de cluster de base de données service base de données relationnelle Amazon

Les instantanés de cluster de base de données Amazon RDS ne disposent pas de politiques basées sur les ressources. Un instantané est partagé via les autorisations du cluster de base de données Amazon RDS. Pour les instantanés de cluster de base de données Amazon RDS, IAM Access Analyzer analyse les listes de contrôle d'accès qui permettent à une entité externe d'accéder à un instantané. Les instantanés de cluster non chiffrés peuvent être publics. Les instantanés de cluster chiffrés ne peuvent pas être partagés publiquement. Les instantanés de cluster chiffrés et non chiffrés et peuvent être partagés avec 20 autres comptes maximum. Pour plus d'informations, consultez Creating a DB Cluster Snapshot (Créer un instantané de cluster de base de données). IAM Access

Analyzer considère la possibilité d'exporter un instantané de cluster de bases de données (par exemple, vers un compartiment Amazon S3) comme un accès sécurisé.

> **Note**
>
> Les résultats d'IAM Access Analyzer n'incluent pas la surveillance d'une partie des clusters et clones de bases de données Amazon RDS utilisés par un autre Compte AWS utilisateur ou une organisation. AWS Resource Access Manager IAM Access Analyzer identifie uniquement les résultats relatifs à l'accès public ou accès-intercompte configuré sur l'instantané du cluster de base de données Amazon RDS.

Référentiels Amazon Elastic Container Registry

Pour les référentiels Amazon ECR, IAM Access Analyzer analyse les politiques basées sur les ressources, y compris les instructions de condition dans une politique, qui permettent à une entité externe d'accéder à un référentiel (comme les autres types de ressources tels que les rubriques Amazon SNS et les systèmes de fichiers d'EFS Amazon SNS). Pour les référentiels Amazon ECR, un principal doit être autorisé à `ecr:GetAuthorizationToken` via une politique basée sur l'identité pour être considéré comme disponible en externe.

Systèmes de fichiers Amazon Elastic File System

Pour les systèmes de fichiers Amazon EFS, IAM Access Analyzer analyse les politiques, notamment les instructions de condition dans une politique, permettant à une entité externe l'accès à un système de fichiers. Un système de fichiers Amazon EFS est accessible de l'extérieur si les responsables d'un compte situé en dehors de votre zone de confiance peuvent effectuer des opérations sur ce système de fichiers. L'accès est défini par une politique de système de fichiers qui utilise IAM et par la manière dont le système de fichiers est monté. Par exemple, le montage de votre système de fichiers Amazon EFS sur un autre compte est considéré comme accessible de l'extérieur, sauf si ce compte appartient à votre organisation et que vous l'ayez définie comme votre zone de confiance. Si vous montez le système de fichiers à partir d'un cloud privé virtuel avec un sous-réseau public, le système de fichiers est accessible de l'extérieur. Lorsque vous utilisez Amazon EFS avec AWS Transfer Family, les demandes d'accès au système de fichiers reçues d'un serveur Transfer Family appartenant à un compte différent de celui du système de fichiers sont bloquées si le système de fichiers autorise l'accès public.

Streams Amazon DynamoDB

IAM Access Analyzer détermine si une politique DynamoDB autorise au moins une action entre comptes permettant à une entité externe d'accéder à un flux DynamoDB. Pour plus d'informations sur les actions entre comptes prises en charge pour DynamoDB, consultez la section Actions IAM prises en charge par des politiques basées sur les ressources dans le manuel du développeur Amazon DynamoDB.

Tables Amazon DynamoDB

IAM Access Analyzer génère une recherche pour une table DynamoDB si une politique DynamoDB autorise au moins une action entre comptes permettant à une entité externe d'accéder à une table ou à un index DynamoDB. Pour plus d'informations sur les actions entre comptes prises en charge pour DynamoDB, consultez la section Actions IAM prises en charge par des politiques basées sur les ressources dans le manuel du développeur Amazon DynamoDB.

Paramètres pour IAM Access Analyzer

Si vous configurez AWS Identity and Access Management Access Analyzer dans votre compte AWS Organizations de gestion, vous pouvez ajouter un compte membre de l'organisation en tant qu'administrateur délégué chargé de gérer IAM Access Analyzer pour votre organisation. L'administrateur délégué dispose des autorisations pour créer et gérer des analyseurs au sein de l'organisation. Seul le compte de gestion peut ajouter un administrateur délégué.

Administrateur délégué pour IAM Access Analyzer

L'administrateur délégué pour l'analyseur d'accès IAM est un compte membre de l'organisation qui dispose d'autorisations pour créer et gérer des analyseurs qui analysent l'accès au sein de l'organisation. Seul le compte de gestion peut ajouter, supprimer ou modifier un administrateur délégué.

Si vous ajoutez un administrateur délégué, vous pouvez ultérieurement passer à un compte différent pour l'administrateur délégué. Dans ce cas, le compte d'administrateur délégué précédent perd l'autorisation sur tous les analyseurs qui ont été créés à l'aide de ce compte pour analyser les accès à travers l'organisation. Ces analyseurs passent à l'état désactivé et ne génèrent plus de résultats et ne mettent pas à jour les résultats existants. Les résultats existants pour ces analyseurs ne sont plus accessibles. Vous pouvez y accéder à nouveau ultérieurement en configurant le compte en tant qu'administrateur délégué. Si vous savez que vous n'utiliserez pas le même compte qu'un

administrateur délégué, vous pouvez envisager de supprimer les analyseurs avant de changer d'administrateur délégué. Cela supprime tous les résultats générés. Lorsque le nouvel administrateur délégué crée de nouveaux analyseurs, de nouvelles instances des mêmes résultats sont générées. Vous ne perdez pas les résultats. Ceux-ci sont simplement générés pour le nouvel analyseur dans un compte différent. Vous pouvez également continuer à accéder aux résultats de l'organisation à l'aide du compte de gestion de l'organisation, qui dispose également des autorisations d'administrateur. Le nouvel administrateur délégué doit créer de nouveaux analyseurs pour que l'IAM Access Analyzer commence à surveiller les ressources de votre organisation.

Si l'administrateur délégué quitte l' AWS organisation, les privilèges d'administration délégués sont supprimés du compte. Tous les analyseurs du compte dont l'organisation est la zone de confiance passent à l'état désactivé. Les résultats existants pour ces analyseurs ne sont plus accessibles.

La première fois que vous configurez des analyseurs dans le compte de gestion, vous pouvez choisir l'option Ajouter un administrateur délégué sur la page Paramètres de l'analyseurdans la console de l'analyseur d'accès IAM.

> **Note**
>
> L'analyseur d'accès IAM facture les analyseurs d'accès non utilisés en fonction du nombre de rôles et d'utilisateurs IAM analysés par analyseur et par mois. Si vous créez un analyseur d'accès non utilisé dans le compte de gestion et le compte d'administrateur délégué, les deux analyseurs d'accès non utilisés vous seront facturés. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

Pour ajouter un administrateur délégué à l'aide de la console

1. Connectez-vous à la AWS console à l'aide du compte de gestion de votre organisation.
2. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
3. Sous Accéder à l'analyseur, sélectionnez Paramètres de l'analyseur.
4. Choisissez Add delegated administrator (Ajouter un administrateur délégué).
5. Dans le champ Administrateur délégué, saisissez le numéro de Compte AWS d'un compte membre de l'organisation à désigner comme administrateur délégué.

   Le compte doit être membre de votre organisation.
6. Sélectionnez Enregistrer les modifications.

Pour ajouter un administrateur délégué à l'aide du AWS CLI ou des AWS SDK

Lorsque vous créez un analyseur pour analyser l'accès à l'ensemble de l'organisation dans un compte d'administrateur délégué à l'aide de la AWS CLI, de l' AWS API (à l'aide AWS des SDK) ou AWS CloudFormation, vous devez utiliser des AWS Organizations API pour activer l'accès aux services pour IAM Access Analyzer et enregistrer le compte membre en tant qu'administrateur délégué.

1. Activez un accès aux services fiable pour IAM Access Analyzer dans. AWS Organizations Consultez la section Comment activer ou désactiver l'accès sécurisé dans le guide de AWS Organizations l'utilisateur.
2. Enregistrez un compte membre valide de votre AWS organisation en tant qu'administrateur délégué à l'aide de l'opération AWS Organizations `RegisterDelegatedAdministrator`API ou de la `register-delegated-administrator` AWS CLI commande.

Après avoir modifié l'administrateur délégué, le nouvel administrateur doit créer des analyseurs pour commencer à surveiller l'accès aux ressources de votre organisation.

Supprimer les analyseurs

Vous pouvez supprimer les analyseurs d’accès externes existants et non utilisés depuis la page des Paramètres de l’analyseur. Lorsque vous supprimez un analyseur, les ressources spécifiées dans l’analyseur ne sont plus surveillées et aucun nouveau résultat n’est généré. Tous les résultats générés par l’analyseur sont supprimés.

Pour les résultats supprimés parce que l'analyseur qui les a générés est supprimé, l'événement est envoyé EventBridge dans les deux jours suivant la suppression de l'analyseur. La suppression des résultats du Security Hub peut prendre jusqu’à 90 jours après la suppression de l’analyseur.

Pour supprimer un analyseur

1. Ouvrez la console IAM à l’adresse https://console.aws.amazon.com/iam/.
2. Sous Accéder à l’analyseur, sélectionnez Paramètres de l’analyseur.
3. Sélectionnez l’analyseur à supprimer, puis choisissez Supprimer.
4. Tapez **`delete`** dans la zone de texte de confirmation, puis choisissez Supprimer.

Règles d'archivage

Les règles d'archivage archivent automatiquement les nouveaux résultats qui répondent aux critères que vous définissez lors de la création de la règle. Vous pouvez également appliquer rétroactivement des règles d'archivage pour archiver des résultats existants répondant aux critères de règles d'archivage. Par exemple, vous pouvez créer une règle d'archivage pour archiver automatiquement tous les résultats pour un compartiment Amazon S3 spécifique auquel vous accordez régulièrement l'accès. Si vous accordez à un principal spécifique l'accès à plusieurs ressources, vous pouvez créer une règle qui archive automatiquement tout nouveau résultat généré pour l'accès accordé à ce principal. Cela vous permet de vous concentrer uniquement sur les résultats actifs qui peuvent indiquer un risque de sécurité.

Lorsque vous créez une règle d'archivage, seuls les nouveaux résultats qui correspondent aux critères de la règle sont automatiquement archivés. Les résultats existants ne sont pas automatiquement archivés. Lorsque vous créez une règle, vous pouvez y inclure jusqu'à 20 valeurs par critère. Pour obtenir la liste des clés de filtre que vous pouvez utiliser pour créer ou mettre à jour une règle d'archivage, veuillez consulter Clés de filtre de l'IAM Access Analyzer.

> **Note**
>
> Lorsque vous créez ou modifiez une règle d'archivage, l'IAM Access Analyzer ne valide pas les valeurs que vous incluez dans le filtre de la règle. Par exemple, si vous ajoutez une règle pour correspondre à un Compte AWS, l'analyseur d'accès IAM accepte toute valeur dans le champ, même s'il ne s'agit pas d'un numéro de compte AWS valide.

Pour créer une règle d'archivage

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Accéder à l'analyseur, puis sélectionnez Paramètres de l'analyseur.
3. Dans la section Analyseurs, choisissez l'analyseur pour lequel vous souhaitez créer une règle d'archivage.
4. Dans l'onglet Règles d'archivage, choisissez Créer une règle d'archivage.
5. Entrez un nom pour la règle si vous souhaitez modifier le nom par défaut.
6. Dans la section Rule (Règle) sous Criteria (Critères), sélectionnez une propriété à faire correspondre pour la règle.

7. Choisissez une condition pour la valeur de la propriété, telle que Contient, Est ou N’est pas égal à.

   Les opérateurs disponibles dépendent de la propriété que vous sélectionnez.
8. Vous pouvez également ajouter des valeurs supplémentaires pour la propriété ou ajouter des critères supplémentaires pour la règle. En ce qui concerne les résultats des accès externes, pour vous assurer que votre règle n’archive pas de nouveaux résultats en accès public, vous pouvez également inclure le critère Accès public et lui attribuer la valeur False.

   Pour ajouter une valeur supplémentaire à un critère, sélectionnez Add another value (Ajouter une autre valeur). Pour ajouter un autre critère à la règle, choisissez Ajouter un critère.
9. Lorsque vous avez terminé d'ajouter des critères et des valeurs, sélectionnez Create rule (Créer une règle) pour appliquer la règle uniquement aux nouveaux résultats. Choisissez Create and archive active findings (Créer et archiver les résultats actifs) pour archiver les résultats nouveaux et existants en fonction des critères de la règle. Dans la section Results (Résultats), vous pouvez consulter la liste des résultats actifs auxquels la règle d'archivage s'applique.

Par exemple, pour créer une règle pour les résultats des accès externes qui archive automatiquement les résultats pour les compartiments Amazon S3 : sélectionnez Type de ressource, puis Est pour la condition. Choisissez ensuite le compartiment S3 dans la liste Valeur.

Pour créer une règle pour les résultats des accès non utilisés qui archive automatiquement tout résultat pour un compte particulier, choisissez Compte propriétaire de la ressource, puis Égal à pour la condition. Entrez l' Compte AWS ID dans la zone de texte Valeur.

Continuez à définir des critères pour personnaliser la règle en fonction de votre environnement, puis sélectionnez Créer une règle.

Si vous créez une règle et ajoutez plusieurs critères, vous pouvez supprimer un critère de la règle en choisissant Remove this criterion (Supprimer ce critère). Vous pouvez supprimer une valeur ajoutée pour un critère en choisissant Remove value (Supprimer la valeur).

Pour modifier une règle d'archivage

1. Choisissez le nom de la règle à modifier dans la colonne Nom.

   Vous ne pouvez modifier qu'une seule règle d'archivage à la fois.
2. Ajoutez les nouveaux critères ou supprimez des critères et des valeurs existants pour chaque critère.

3. Choisissez Enregistrer les modifications pour appliquer la règle uniquement aux nouveaux résultats. Choisissez Enregistrer et archiver les résultats actifs pour archiver les résultats nouveaux et existants en fonction des critères de la règle.

Pour supprimer une règle d'archivage

1. Cochez la case correspondant aux règles que vous souhaitez supprimer.
2. Sélectionnez Delete (Supprimer).
3. Tapez **`delete`** dans la boîte de dialogue de confirmation Supprimer une règle d'archivage puis sélectionnez Supprimer.

Les règles sont supprimées uniquement de l'analyseur dans la région actuelle. Vous devez supprimer les règles d'archivage séparément pour chaque analyseur que vous avez créé dans d'autres régions.

Surveillance AWS Identity and Access Management Access Analyzer avec Amazon EventBridge

Utilisez les informations de cette rubrique pour savoir comment surveiller les résultats d'IAM Access Analyzer et accéder aux aperçus avec Amazon. EventBridge EventBridge est la nouvelle version d'Amazon CloudWatch Events.

Événements de résultats

IAM Access Analyzer envoie un événement EventBridge pour chaque résultat généré, pour une modification du statut d'un résultat existant et lorsqu'un résultat est supprimé. Pour recevoir les résultats et les notifications concernant les résultats, vous devez créer une règle d'événement dans Amazon EventBridge. Lorsque vous créez une règle d'événement, vous pouvez également spécifier une action cible à déclencher en fonction de la règle. Par exemple, vous pouvez créer une règle d'événement qui déclenche une rubrique Amazon SNS lorsqu'un événement pour un nouveau résultat est reçu en provenance de l’IAM Access Analyzer

Événements de prévisualisation de l'accès

IAM Access Analyzer envoie un événement pour chaque aperçu de l'accès et EventBridge pour chaque modification de son statut. Cela inclut un événement lorsque la prévisualisation de l'accès est initialement créée (statut Création), lorsque la prévisualisation de l'accès est terminée (statut Terminé) ou lorsque la création de la prévisualisation de l'accès a échoué (statut Échec). Pour

recevoir des notifications concernant les aperçus d'accès, vous devez créer une règle d'événement dans EventBridge. Lorsque vous créez une règle d'événement, vous pouvez spécifier une action cible à déclencher en fonction de la règle. Par exemple, vous pouvez créer une règle d'événement qui déclenche une rubrique Amazon SNS lorsqu'un événement pour une nouvelle prévisualisation est reçue en provenance de l'IAM Access Analyzer

Fréquence de notification d'événement

IAM Access Analyzer envoie des événements pour les nouvelles découvertes et les nouvelles découvertes avec des mises à jour de statut EventBridge dans l'heure qui suit la survenue de l'événement dans votre compte. IAM Access Analyzer envoie également des événements EventBridge lorsqu'un résultat résolu est supprimé en raison de l'expiration de la période de rétention. Pour les résultats supprimés parce que l'analyseur qui les a générés est supprimé, l'événement est envoyé EventBridge environ 24 heures après la suppression de l'analyseur. Lorsqu'un résultat est supprimé, le statut du résultat n'est pas modifié. En revanche, l'attribut `isDeleted` est défini sur `true`. IAM Access Analyzer envoie également des événements pour les aperçus d'accès nouvellement créés et les modifications du statut des aperçus d'accès à. EventBridge

Exemples d'événements liés aux résultats des accès externes

Voici un exemple d'événement de recherche d'accès externe IAM Access Analyzer envoyé à. EventBridge La `id` liste indique l'identifiant de l'événement dans EventBridge. Pour en savoir plus, consultez la section Événements et modèles d'événements dans EventBridge.

Dans l'objet `detail`, les valeurs des attributs `accountId` et `region` se réfèrent au compte et à la région indiqués dans le résultat. L'attribut `isDeleted` indique si l'événement provient du résultat en cours de suppression. L'attribut `id` représente l'ID de résultat. Le tableau `resources` est un singleton avec l'ARN de l'analyseur qui a généré le résultat.

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "action": [
            "s3:GetObject"
        ],
        "analyzedAt": "2019-11-21T01:22:22Z",
        "condition": {},
        "createdAt": "2019-11-20T04:58:50Z",
```

```
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "isPublic": false,
        "principal": {
            "AWS": "999988887777"
        },
        "region": "us-west-2",
        "resource": "arn:aws:s3:::my-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```

IAM Access Analyzer envoie également des événements EventBridge pour détecter les erreurs. Un résultat d'erreur est un résultat généré lorsque l'IAM Access Analyzer ne peut pas analyser la ressource. Les événements pour les résultats d'erreur incluent un attribut `error` comme illustré dans l'exemple suivant.

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "analyzedAt": "2019-11-21T01:22:22Z",
        "createdAt": "2019-11-20T04:58:50Z",
        "error": "ACCESS_DENIED",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "region": "us-west-2",
        "resource": "arn:aws:s3:::my-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
```

```
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```

Exemples d'événements liés aux résultats des accès non utilisés

Voici un exemple d'événement de recherche d'accès non utilisé d'IAM Access Analyzer envoyé à. EventBridge La `id` liste indique l'identifiant de l'événement dans EventBridge. Pour en savoir plus, consultez la section Événements et modèles d'événements dans EventBridge.

Dans l'objet `detail`, les valeurs des attributs `accountId` et `region` se réfèrent au compte et à la région indiqués dans le résultat. L'attribut `isDeleted` indique si l'événement provient du résultat en cours de suppression. L'attribut `id` représente l'ID de résultat.

```
{
    "version": "0",
    "id": "dc7ce3ee-114b-3243-e249-7f10f9054b21",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
    "account": "123456789012",
    "time": "2023-09-29T17:31:40Z",
    "region": "us-west-2",
    "resources": [
       "arn:aws:access-analyzer:us-west-2:123456789012:analyzer/
integTestLongLivingAnalyzer-DO-NOT-DELETE"
       ],
    "detail": {
        "findingId": "b8ae0460-5d29-4922-b92a-ba956c986277",
        "resource": "arn:aws:iam::111122223333:role/FindingIntegTestFakeRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "111122223333",
        "createdAt": "2023-09-29T17:29:18.758Z",
        "updatedAt": "2023-09-29T17:29:18.758Z",
```

```
        "analyzedAt": "2023-09-29T17:29:18.758Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "62160bda-8e94-46d6-ac97-9670930d8ffb",
        "isDeleted": false,
        "findingType": "UnusedPermission",
        "numberOfUnusedServices": 0,
        "numberOfUnusedActions": 1
        }
    }
```

IAM Access Analyzer envoie également des événements EventBridge pour détecter les erreurs. Un résultat d'erreur est un résultat généré lorsque l'IAM Access Analyzer ne peut pas analyser la ressource. Les événements pour les résultats d'erreur incluent un attribut `error` comme illustré dans l'exemple suivant.

```
{
    "version": "0",
    "id": "c2e7aa1a-4df7-7652-f33e-64113b8997d4",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
    "account": "111122223333",
    "time": "2023-10-31T20:26:12Z",
    "region": "us-west-2",
    "resources": [
      "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/ba811f91-
de99-41a4-97c0-7481898b53f2"
      ],
    "detail": {
        "findingId": "b01a34f2-e118-46c9-aef8-0d8526b495c7",
        "resource": "arn:aws:iam::123456789012:role/TestRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "444455556666",
        "createdAt": "2023-10-31T20:26:08.647Z",
        "updatedAt": "2023-10-31T20:26:09.245Z",
        "analyzedAt": "2023-10-31T20:26:08.525Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "7c7a72a2-7963-4c59-ac71-f0be597010f7",
        "isDeleted": false,
        "findingType": "UnusedIAMRole",
        "error": "INTERNAL_ERROR"
        }
```

```
  }
```

Exemple d'événement de prévisualisation de l'accès

L'exemple suivant montre les données du premier événement envoyé EventBridge lorsque vous créez un aperçu des accès. Le tableau `resources` est un singleton avec l'ARN de l'analyseur auquel la prévisualisation de l'accès est associée. Dans l'objet `detail`, l'`id` fait référence à l'ID de prévisualisation de l'accès et `configuredResources` fait référence à la ressource pour laquelle la prévisualisation de l'accès a été créée. Le `status` est `Creating` et fait référence au statut de la prévisualisation de l'accès. Le `previousStatus` n'est pas spécifié car la prévisualisation de l'accès vient juste d'être créée.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::example-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "region": "us-west-2",
        "status": "CREATING",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "aaaabbbb-2222-3333-4444-555566667777",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

L'exemple suivant montre les données d'un événement envoyé à EventBridge pour un aperçu de l'accès avec un changement de statut de `Creating` à`Completed`. Dans l'objet détail, l'ID `id` fait référence à l'ID de prévisualisation de l'accès. Le `status` et le `previousStatus` font référence au statut de la prévisualisation de l'accès, où le statut précédent était `Creating` et où le statut actuel est `Completed`.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::example-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.000Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "COMPLETED",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "11112222-3333-4444-5555-666677778888",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

L'exemple suivant montre les données d'un événement envoyé à EventBridge pour un aperçu de l'accès avec un changement de statut de `Creating` à`Failed`. Dans l'objet `detail`, l'`id` fait référence à l'ID de prévisualisation de l'accès. Le `status` et le `previousStatus` font référence au statut de la prévisualisation de l'accès, où le statut précédent était `Creating` et où le statut actuel est `Failed`. Le champ `statusReason` fournit le code raison indiquant que la prévisualisation de l'accès a échoué en raison d'une configuration de ressource non valide.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::example-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "FAILED",
```

```
        "statusReason": {
            "code": "INVALID_CONFIGURATION"
        },
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "99998888-7777-6666-5555-444433332222",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

Création d'une règle d'événement à l'aide de la console

La procédure suivante explique comment créer une règle d'événement à l'aide de la console.

1. Ouvrez la EventBridge console Amazon à l'adresse https://console.aws.amazon.com/events/.
2. À l'aide des valeurs suivantes, créez une EventBridge règle qui surveille la recherche d'événements ou l'accès aux événements de prévisualisation :

   - Pour Type de règle, choisissez Règle avec un modèle d’événement.
   - Pour Event source (Source de l'événement), choisissez Other (Autres).
   - Pour Event pattern (Modèle d'événement), choisissez Custom patterns (JSON editor) (Modèle personnalisé [éditeur JSON]), puis collez l'un des exemples de modèle d'événement suivant dans la zone de texte :
     - Pour créer une règle basée sur un événement de résultats des accès externes ou non utilisés, utilisez l’exemple de modèle suivant :

       ```
       {
         "source": [
           "aws.access-analyzer"
         ],
         "detail-type": [
           "Access Analyzer Finding"
         ]
       }
       ```

- Pour créer une règle basée uniquement sur un événement de recherche d'accès non utilisé, utilisez l'exemple de modèle suivant :

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Unused Access Finding for IAM entities"
  ]
}
```

> **Note**
>
> Vous ne pouvez pas créer une règle basée uniquement sur un événement de recherche d'accès externe.

- Pour créer une règle basée sur un événement de prévisualisation de l'accès, utilisez l'exemple de modèle suivant :

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Preview State Change"
  ]
}
```

- Pour les types de cibles, choisissez un AWS service, et pour Sélectionner une cible, choisissez une cible telle qu'un sujet ou AWS Lambda une fonction Amazon SNS. La cible est déclenchée lorsqu'un événement correspond au modèle d'événement défini dans la règle est reçu.

Pour en savoir plus sur la création de règles, consultez la section Création de EventBridge règles Amazon qui réagissent aux événements dans le guide de EventBridge l'utilisateur Amazon.

Création d'une règle d'événement à l'aide de la CLI

1. Utilisez ce qui suit pour créer une règle pour Amazon EventBridge à l'aide du AWS CLI. Remplacez le nom de la règle *TestRule*par le nom de votre règle.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"]}"
```

2. Vous pouvez personnaliser la règle pour déclencher des actions cibles uniquement pour un sous-ensemble de résultats générés, tels que les résultats avec des attributs spécifiques. L'exemple suivant montre comment créer une règle qui déclenche une action cible uniquement pour les résultats ayant un statut actif.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"],\"detail-type\":[\"Access Analyzer Finding\"],\"detail\":{\"status\":
[\"ACTIVE\"]}}"
```

   L'exemple suivant montre comment créer une règle qui déclenche une action cible uniquement pour les prévisualisations d'accès ayant un statut qui passe de `Creating` à `Completed`.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"],\"detail-type\":[\"Access Preview State Change\"],\"detail\":{\"status
\":[\"COMPLETED\"]}}"
```

3. Pour définir une fonction Lambda en tant que cible pour la règle que vous avez créée, utilisez l'exemple de commande suivant. Remplacez la région et le nom de la fonction dans l'ARN en fonction de votre environnement.

```
aws events put-targets --rule TestRule --targets Id=1,Arn=arn:aws:lambda:us-
east-1:111122223333:function:MyFunction
```

4. Ajoutez les autorisations requises pour appeler la cible de la règle. L'exemple suivant montre comment accorder des autorisations à une fonction Lambda, en suivant les exemples précédents.

```
aws lambda add-permission --function-name MyFunction --statement-id 1 --action
 'lambda:InvokeFunction' --principal events.amazonaws.com
```

Intégrez Access Analyzer à AWS Security Hub

AWS Security Hubvous fournit une vue complète de l'état de votre sécurité AWS et vous aide à vérifier que votre environnement est conforme aux normes et aux meilleures pratiques du secteur de la sécurité. Security Hub collecte des données de sécurité provenant de AWS comptes, de services et de produits partenaires tiers pris en charge et vous aide à analyser les tendances en matière de sécurité et à identifier les problèmes de sécurité les plus prioritaires.

Lorsque vous AWS Identity and Access Management Access Analyzer intégrez Security Hub, vous pouvez envoyer les résultats d'IAM Access Analyzer à Security Hub. Security Hub peut ensuite inclure ces résultats dans son analyse de votre posture de sécurité.

Table des matières

- Comment l’analyseur d’accès IAM envoie les résultats à Security Hub
  - Types de résultats envoyés par l’analyseur d’accès IAM
  - Latence pour l'envoi des résultats
  - Réessayer lorsque Security Hub n'est pas disponible
  - Mise à jour des résultats existants dans Security Hub
- Affichage des résultats de l’analyseur d’accès IAM dans Security Hub
  - Interprétation des noms de résultats de l’analyseur d’accès IAM dans Security Hub
- Résultats typiques de l’analyseur d’accès IAM
- Activation et configuration de l'intégration
- Comment arrêter l'envoi des résultats

Comment l’analyseur d’accès IAM envoie les résultats à Security Hub

Dans Security Hub, les problèmes de sécurité sont suivis en tant que findings. (résultats) Certains résultats proviennent de problèmes détectés par d'autres AWS services ou par des partenaires tiers. Security Hub utilise également un ensemble de règles pour détecter les problèmes de sécurité et générer des résultats.

Security Hub fournit des outils permettant de gérer les résultats provenant de toutes ces sources. Vous pouvez afficher et filtrer les listes de résultats et afficher les informations sur un résultat. Veuillez consulter Viewing findings (Affichage des résultats)dans le Guide de l'utilisateur AWS Security Hub . Vous pouvez également suivre le statut d'une analyse dans un résultat. Consultez Prendre des mesures à la suite des résultats dans le Guide de l'utilisateur AWS Security Hub .

Tous les résultats de Security Hub utilisent un format JSON standard appelé AWS Security Finding Format (ASFF). Le format ASFF comprend des informations sur la source du problème, les ressources affectées et le statut actuel du résultat. Consultez AWS Security Finding Format (ASFF) dans le Guide de l'utilisateur AWS Security Hub .

AWS Identity and Access Management Access Analyzer est l'un des AWS services qui envoie les résultats à Security Hub. Pour les accès non utilisés, IAM Access Analyzer détecte les accès non utilisés accordés aux utilisateurs ou aux rôles IAM et génère un résultat pour chacun d'entre eux. IAM Access Analyzer envoie ensuite ces résultats à Security Hub. Pour l'accès externe, IAM Access Analyzer détecte une déclaration de politique qui autorise l'accès public ou intercomptes aux principaux externes sur une ressource prise en charge dans votre organisation ou votre compte. IAM Access Analyzer génère une recherche d'accès public, qu'il envoie ensuite à Security Hub. Pour l'accès entre comptes, IAM Access Analyzer envoie une seule constatation pour un principal externe à la fois à Security Hub. S'il existe plusieurs résultats entre comptes dans IAM Access Analyzer, vous devez résoudre le résultat du Security Hub pour le principal externe unique avant qu'IAM Access Analyzer ne fournisse le résultat croisé suivant. Pour obtenir la liste complète des principaux externes disposant d'un accès entre comptes en dehors de la zone de confiance de l'analyseur, vous devez consulter les résultats dans IAM Access Analyzer.

Types de résultats envoyés par l’analyseur d’accès IAM

L’analyseur d’accès IAM envoie les résultats à Security Hub à l’aide du Format de recherche de sécuritéAWS (ASFF). Dans le format ASFF, le champ `Types` fournit le type de résultat. Les résultats de l’analyseur d’accès IAM peuvent avoir les valeurs suivantes pour `Types`.

- Résultats d’accès externes : effets/exposions de données/accès externe accordé
- Constatations relatives à l'accès externe — Contrôles du logiciel et de la configuration/Meilleures pratiques en AWS matière de sécurité/Accès externe accordé
- Résultats relatifs aux accès non utilisés — Vérifications du logiciel et de la configuration/Bonnes pratiques en matière de AWS sécurité/Autorisation non utilisée
- Résultats relatifs aux accès non utilisés — Contrôles du logiciel et de la configuration/Meilleures pratiques en matière de AWS sécurité/Rôle IAM inutilisé
- Résultats relatifs aux accès non utilisés — Contrôles du logiciel et de la configuration/Meilleures pratiques en matière de AWS sécurité/Mot de passe utilisateur IAM non utilisé
- Résultats d'accès non utilisés — Contrôles du logiciel et de la configuration/Meilleures pratiques en matière de AWS sécurité/Clé d'accès utilisateur IAM non utilisée

Latence pour l'envoi des résultats

Lorsque l’analyseur d’accès IAM crée un nouveau résultat, ce dernier est généralement envoyé à Security Hub dans les 30 minutes qui suivent. Dans de rares cas et sous certaines conditions, l’analyseur d’accès IAM n’est pas avisé de l’ajout ou de la mise à jour d’une politique. Par exemple, une modification des paramètres de blocage de l'accès public Amazon S3 au niveau compte peut prendre jusqu'à 12 heures. En outre, en cas de problème de livraison lié à la livraison du AWS CloudTrail journal, le changement de politique ne déclenche pas une nouvelle analyse de la ressource signalée dans le résultat. Lorsque cela se produit, l’analyseur d’accès IAM analyse la politique nouvelle ou mise à jour au cours de l’analyse périodique suivante.

Réessayer lorsque Security Hub n'est pas disponible

Si Security Hub n’est pas disponible, l’analyseur d’accès IAM tente de nouveau d’envoyer les résultats périodiquement.

Mise à jour des résultats existants dans Security Hub

Après avoir envoyé un résultat à Security Hub, il AWS Identity and Access Management Access Analyzer envoie des mises à jour pour refléter les observations supplémentaires concernant l'activité de recherche à Security Hub. Les mises à jour sont reflétées dans le même résultat.

Étant donné que l’analyseur d’accès IAM regroupe les résultats d’accès externes par ressource, le résultat d’une ressource dans Security Hub est actif si au moins un des résultats de la ressource dans l’analyseur d’accès IAM est actif. Si toutes les résultats de l’analyseur d’accès IAM pour une ressource sont archivés ou résolus, le résultat de Security Hub est archivé. Le résultat Security Hub est mise à jour lorsque vous modifiez l'accès de la politique entre l'accès public et l'accès entre comptes. Cette mise à jour peut inclure des modifications du type, du titre, de la description et de la gravité du résultat.

L’analyseur d’accès IAM ne regroupe pas les résultats des accès non utilisés par ressource. Ainsi, si un résultat des accès non utilisés est résolu dans l’analyseur d’accès IAM, le résultat de Security Hub est résolu. Le résultat Security Hub est mis à jour lorsque vous mettez à jour l’utilisateur ou le rôle IAM qui a généré le résultat d’accès non utilisé.

Affichage des résultats de l’analyseur d’accès IAM dans Security Hub

Pour afficher vos résultats analyseur d’accès IAM dans Security Hub, sélectionnez Voir les résultats dans la section AWS : analyseur d’accès IAM de la page récapitulative. Vous pouvez également

choisir Findings (Résultats) dans le panneau de navigation. Vous pouvez ensuite filtrer les résultats pour n'afficher que AWS Identity and Access Management Access Analyzer les résultats en choisissant le champ Nom du produit : avec une valeur de**`IAM Access Analyzer`**.

Interprétation des noms de résultats de l’analyseur d’accès IAM dans Security Hub

AWS Identity and Access Management Access Analyzer envoie les résultats à Security Hub en utilisant le format ASFF (AWS Security Finding Format). Dans ASFF, le champ Types fournit le type de recherche. Les types ASFF utilisent un schéma de dénomination différent de AWS Identity and Access Management Access Analyzer. Le tableau suivant contient des informations détaillées sur tous les types ASFF associés aux AWS Identity and Access Management Access Analyzer résultats tels qu'ils apparaissent dans Security Hub.

Type de résultat ASFF	Titre de résultat dans Security Hub	Description
Effets/Exposion de données/Accès externe accordé	<ARN de la ressource >permet l'accès public	Une politique basée sur les ressources attachée à la ressource permet à touts les principaux externes d'accéder à la ressource.
Vérifications du logiciel et de la configuration/Bonnes pratiques AWS de sécurité/ Accès externe accordé	<ARN de la ressource >permet l'accès entre comptes	Une politique basée sur les ressources attachée à la ressource fournit un accès entre comptes aux principaux externes en dehors de la zone confiance de l'analyseur .
Vérifications du logiciel et de la configuration/Bonnes pratiques de AWS sécurité/ Autorisations non utilisées	<resource ARN> contient des autorisations non utilisées	Un utilisateur ou un rôle contient des autorisations de service et d’action non utilisés.
Contrôles du logiciel et de la configuration/Meilleures pratiques en AWS matière de sécurité/Rôle IAM inutilisé	<resource ARN> contient un rôle IAM non utilisé	Un utilisateur ou un rôle contient un rôle IAM non utilisé.

Type de résultat ASFF	Titre de résultat dans Security Hub	Description
Contrôles du logiciel et de la configuration/Meilleures pratiques en AWS matière de sécurité/Mot de passe utilisateur IAM non utilisé	<resource ARN> contient un mot de passe utilisateur IAM non utilisé	Un utilisateur ou un rôle contient un mot de passe utilisateur IAM non utilisé.
Contrôles du logiciel et de la configuration/Meilleures pratiques en AWS matière de sécurité/Clé d'accès utilisateur IAM non utilisée	<resource ARN> contient la clé d'accès utilisateur IAM non utilisée	Un utilisateur ou un rôle contient une clé d'accès utilisateur IAM non utilisée.

Résultats typiques de l'analyseur d'accès IAM

L'analyseur d'accès IAM envoie les résultats à Security Hub à l'aide du Format de recherche de sécuritéAWS (ASFF).

Voici un exemple de résultat typique de l'analyseur d'accès IAM pour les résultats d'accès externes.

```
{
    "SchemaVersion": "2018-10-08",
    "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/my-analyzer/
arn:aws:s3:::my-bucket",
    "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
    "GeneratorId": "aws/access-analyzer",
    "AwsAccountId": "111122223333",
    "Types": ["Software and Configuration Checks/AWS Security Best Practices/External
 Access Granted"],
    "CreatedAt": "2020-11-10T16:17:47Z",
    "UpdatedAt": "2020-11-10T16:43:49Z",
    "Severity": {
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
    },
    "Title": "AwsS3Bucket/arn:aws:s3:::my-bucket/ allows cross-account access",
```

```
    "Description": "AWS::S3::Bucket/arn:aws:s3:::my-bucket/ allows cross-account access
 from AWS 444455556666",
    "Remediation": {
        "Recommendation": {"Text": "If the access isn't intended, it indicates a
 potential security risk. Use the console for the resource to modify or remove the
 policy that grants the unintended access. You can use the Rescan button on the Finding
 details page in the Access Analyzer console to confirm whether the change removed the
 access. If the access is removed, the status changes to Resolved."}
    },
    "SourceUrl": "https://console.aws.amazon.com/access-analyzer/home?region=us-
west-2#/findings/details/dad90d5d-63b4-6575-b0fa-ef9c556ge798",
    "Resources": [
        {
            "Type": "AwsS3Bucket",
            "Id": "arn:aws:s3:::my-bucket",
            "Details": {
                "Other": {
                    "External Principal Type": "AWS",
                    "Condition": "none",
                    "Action Granted": "s3:GetObject,s3:GetObjectVersion",
                    "External Principal": "444455556666"
                }
            }
        }
    ],
    "WorkflowState": "NEW",
    "Workflow": {"Status": "NEW"},
    "RecordState": "ACTIVE"
}
```


Voici un exemple de résultat typique de l'analyseur d'accès IAM pour les résultats des accès non utilisés.


```
{
    "Findings": [
    {
      "SchemaVersion": "2018-10-08",
      "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-
DO-NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
      "ProductName": "IAM Access Analyzer",
      "CompanyName": "AWS",
      "Region": "us-west-2",
```

```
      "GeneratorId": "aws/access-analyzer",
      "AwsAccountId": "111122223333",
      "Types": [
        "Software and Configuration Checks/AWS Security Best Practices/Unused
 Permission"
      ],
      "CreatedAt": "2023-09-18T16:29:09.657Z",
      "UpdatedAt": "2023-09-21T20:39:16.651Z",
      "Severity": {
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
      },
      "Title": "AwsIamRole/arn:aws:iam::111122223333:role/IsengardRole-DO-NOT-DELETE/
 contains unused permissions",
      "Description": "AWS::IAM::Role/arn:aws:iam::111122223333:role/IsengardRole-DO-
NOT-DELETE/ contains unused service and action-level permissions",
      "Remediation": {
        "Recommendation": {
          "Text":"If the unused permissions aren’t required, delete the permissions to
 refine access to your account. Use the IAM console to modify or remove the policy that
 grants the unused permissions. If all the unused permissions are removed, the status
 of the finding changes to Resolved."
        }
      },
      "SourceUrl": "https://us-west-2.console.aws.amazon.com/access-analyzer/
home?region=us-west-2#/unused-access-findings?resource=arn%3Aaws%3Aiam%3A
%3A903798373645%3Arole%2FTestRole",
      "ProductFields": {
      "numberOfUnusedActions": "256",
      "numberOfUnusedServices": "15",
      "resourceOwnerAccount": "111122223333",
      "findingId": "DEMO24d8d-0d3f-4d3d-99f4-299fc8a62ee7",
      "findingType": "UnusedPermission",
      "aws/securityhub/FindingId": "arn:aws:securityhub:us-west-2::product/aws/access-
analyzer/arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-DO-
NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "aws/securityhub/ProductName": "AM Access Analyzer",
      "aws/securityhub/CompanyName": "AWS"
    },
    "Resources": [
    {
      "Type": "AwsIamRole",
      "Id": "arn:aws:iam::111122223333:role/TestRole"
```

```
    }
  ],
  "WorkflowState": "NEW",
  "Workflow": {
    "Status": "NEW"
    },
  "RecordState": "ARCHIVED",
  "FindingProviderFields": {
    "Severity": {
      "Label": "LOW"
    },
    "Types": [
    "Software and Configuration Checks/AWS Security Best Practices/Unused Permission"
    ]
  }
  }
]
}
```

Activation et configuration de l'intégration

Pour utiliser l'intégration avec Security Hub, vous devez activer Security Hub. Pour plus d'informations sur la façon d'activer Security Hub, veuillez consulter Configuration de Security Hub dans le Guide de l'utilisateur AWS Security Hub .

Lorsque vous activez à la fois l'analyseur d'accès IAM et Security Hub, l'intégration est activée automatiquement. L'analyseur d'accès IAM commence immédiatement à envoyer les résultats à Security Hub.

Comment arrêter l'envoi des résultats

Pour arrêter l'envoi des résultats à Security Hub, vous pouvez utiliser la console Security Hub ou l'API.

Veuillez consulter Désactivation et activation du flux de résultats à partir d'une intégration (console) ou Désactivation du flux de résultats d'une intégration (API Security Hub, AWS CLI) dans le Guide de l'utilisateur AWS Security Hub .

Journalisation des appels d'API IAM Access Analyzer avec AWS CloudTrail

IAM Access Analyzer est intégré à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans IAM Access Analyzer.

CloudTrail capture tous les appels d'API pour IAM Access Analyzer sous forme d'événements. Les appels enregistrés contiennent les appels de la console IAM Access Analyzer ainsi que les appels de code aux opérations d'API IAM Access Analyzer.

Si vous créez un suivi, vous pouvez activer la diffusion continue d' CloudTrail événements vers un compartiment Amazon S3, y compris des événements pour IAM Access Analyzer. Si vous ne configurez pas de suivi, vous pouvez toujours consulter les événements les plus récents dans la CloudTrail console dans Historique des événements.

À l'aide des informations collectées par CloudTrail, vous pouvez déterminer la demande envoyée à IAM Access Analyzer, l'adresse IP à partir de laquelle la demande a été faite, l'auteur de la demande, la date à laquelle elle a été faite et des informations supplémentaires.

Pour en savoir plus CloudTrail, consultez le guide de AWS CloudTrail l'utilisateur.

Informations sur l'analyseur d'accès IAM dans CloudTrail

CloudTrail est activé sur votre AWS compte lorsque vous le créez. Lorsqu'une activité se produit dans IAM Access Analyzer, cette activité est enregistrée dans un CloudTrail événement avec d'autres événements de AWS service dans l'historique des événements. Vous pouvez consulter, rechercher et télécharger les événements récents dans votre AWS compte. Pour plus d'informations, consultez la section Affichage des événements avec l'historique des CloudTrail événements.

Pour un enregistrement continu des événements de votre AWS compte, y compris des événements pour IAM Access Analyzer, créez un suivi. Un suivi permet CloudTrail de fournir des fichiers journaux à un compartiment Amazon S3. Par défaut, lorsque vous créez un parcours dans la console, celui-ci s'applique à toutes les AWS régions. Le journal enregistre les événements de toutes les régions de la AWS partition et transmet les fichiers journaux au compartiment Amazon S3 que vous spécifiez. En outre, vous pouvez configurer d'autres AWS services pour analyser plus en détail les données d'événements collectées dans les CloudTrail journaux et agir en conséquence. Pour plus d’informations, consultez les ressources suivantes :

- Vue d’ensemble de la création d’un journal d’activité
- CloudTrail Services et intégrations pris en charge
- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers CloudTrail journaux de plusieurs régions et réception de fichiers CloudTrail journaux de plusieurs comptes

Toutes les actions d'IAM Access Analyzer sont enregistrées CloudTrail et documentées dans la référence d'API d'IAM Access Analyzer. Par exemple, les appels au `CreateAnalyzer` `CreateArchiveRule` et les `ListFindings` actions génèrent des entrées dans les fichiers CloudTrail journaux.

Chaque événement ou entrée de journal contient des informations sur la personne ayant initié la demande. Les informations relatives à l'identité permettent de déterminer les éléments suivants :

- Si la demande a été faite avec les informations d'identification de l'utilisateur root ou AWS Identity and Access Management (IAM).
- Si la demande a été effectuée avec les informations d'identification de sécurité temporaires d'un rôle ou d'un utilisateur fédéré.
- Si la demande a été faite par un autre AWS service.

Pour plus d'informations, consultez l'élément CloudTrail UserIdentity.

Présentation des entrées des fichiers journaux IAM Access Analyzer

Un suivi est une configuration qui permet de transmettre des événements sous forme de fichiers journaux à un compartiment Amazon S3 que vous spécifiez. CloudTrail les fichiers journaux contiennent une ou plusieurs entrées de journal. Un événement représente une demande unique provenant de n'importe quelle source et inclut des informations sur l'action demandée, la date et l'heure de l'action, les paramètres de la demande, etc. CloudTrail les fichiers journaux ne constituent pas une trace ordonnée des appels d'API publics, ils n'apparaissent donc pas dans un ordre spécifique.

L'exemple suivant montre une entrée de CloudTrail journal qui illustre l'`CreateAnalyzer`opération effectuée par une session assumée nommée `Alice-tempcreds` le « 14 juin 2021 ». La séance de rôle a été générée par le rôle nommé `admin-tempcreds`.

```
{
  "eventVersion": "1.05",
  "userIdentity":   {
    "type": "AssumedRole",
    "principalId": "AROAIBKEVSQ6C2EXAMPLE:Alice-tempcreds",
    "arn": "arn:aws:sts::111122223333:assumed-role/admin-tempcreds/Alice-tempcreds",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
```

```
      "attributes": {
        "mfaAuthenticated": "true",
        "creationDate": "2021-06-14T22:54:20Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:iam::111122223333:role/admin-tempcreds",
        "accountId": "111122223333",
        "userName": "admin-tempcreds"
      },
     "webIdFederationData": {},
    }
  },
  "eventTime": "2021-06-14T22:57:36Z",
  "eventSource": "access-analyzer.amazonaws.com",
  "eventName": "CreateAnalyzer",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "198.51.100.179",
  "userAgent": "aws-sdk-java/1.12.79 Linux/5.4.141-78.230 OpenJDK_64-
Bit_Server_VM/25.302-b08 java/1.8.0_302 vendor/Oracle_Corporation cfg/retry-mode/
standard",
  "requestParameters": {
    "analyzerName": "test",
    "type": "ACCOUNT",
    "clientToken": "11111111-abcd-2222-abcd-222222222222",
        "tags": {
            "tagkey1": "tagvalue1"
        }
  },
  "responseElements": {
    "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/test"
  },
  "requestID": "22222222-dcba-4444-dcba-333333333333",
  "eventID": "33333333-bcde-5555-bcde-444444444444",
  "readOnly": false,
  "eventType": "AwsApiCall",,
  "managementEvent": true,
  "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Clés de filtre de l'IAM Access Analyzer

Vous pouvez utiliser les touches de filtre ci-dessous pour définir une règle d'archivage (`CreateArchiveRule`), mettre à jour une règle d'archivage (`UpdateArchiveRule`), récupérer une liste de résultats (`ListFindings`et `ListFindingsV2`) ou récupérer une liste de résultats d'aperçu d'accès pour une ressource (`ListAccessPreviewFindings`). Il n'y a aucune différence entre l'utilisation de l'API IAM et AWS CloudFormation la configuration des règles d'archivage.

Criterion	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès
ressource	ARN identifiant de manière unique la ressource à laquelle le principal externe a accès. Pour en savoir plus, veuillez consulter Amazon Resource Names (ARN).	Chaîne	 Oui	 Oui	 Oui
resourceType `AWS::IAM::Role | AWS::KMS::Key | AWS::Lambda::Function | AWS::Lambda::LayerVersion`	Type de ressource auquel le principal externe a accès.	Chaîne	 Oui	 Oui	 Oui

Criterion	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès
`| AWS::S3::Bucket | AWS::S3Express::DirectoryBucket | AWS::SQS::Queue | AWS::SecretsManager::Secret | AWS::EFS::FileSystem | AWS::EC2::Snapshot | AWS::ECR::Repository | AWS::RDS::DBSnapshot | AWS::RDS::DBClusterSnapshot | AWS::SNS:`					

Criterion	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès
`:Topic | AWS::DynamoDB::Stream | AWS::DynamoDB::Table`					
resourceOwnerAccount	L'identifiant de AWS compte à 12 chiffres propriétaire de la ressource. Pour en savoir plus, consultez Identifiants de compte AWS.	Chaîne	 Oui	 Oui	 Oui
isPublic	Indique si le résultat signale une ressource dotée d'une politique autorisant l'accès public.	Booléen	 Oui	 Oui	 Oui

Criterion	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès
Type de recherche `UnusedIAMRole | UnusedIAMUserAccessKey | UnusedIAMUserPassword | UnusedPermission`	Type du résultat . Vous pouvez uniquement filtrer par type de recherche pour les résultats d'accès non utilisés.	Chaîne	 Oui	 Oui	 Oui
status `ACTIVE | ARCHIVED | RESOLVED`	Statut actuel du résultat.	Chaîne	 Non	 Oui	 Oui
error	Indique l'erreur signalée pour le résultat.	Chaîne	 Oui	 Oui	 Oui

Criterion	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès
principal.AWS	Le compte a accordé l'accès à la ressource dans le champ `Principal` du résultat. Entrez l'ID de AWS compte à 12 chiffres ou l'ARN de l' AWS utilisateur ou du rôle externe. Pour en savoir plus, consultez Identifiants de compte AWS.	Chaîne	 Oui	 Oui	 Oui
principal.Federated	ARN de l'identité fédérée qui a accès à la ressource dans le résultat. Pour en savoir plus, veuillez consulter Fournisseurs d'identité et fédération.	Chaîne	 Oui	 Oui	 Oui
condition.aws : PrincipalArn	ARN du principal (utilisateur, rôle ou groupe IAM) indiquée comme condition d'accès aux ressources. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	Chaîne	 Oui	 Oui	 Oui

Criterion	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès
condition .aws : ID Principal Org	Identifiant de l'organisation du principal indiqué comme condition d'accès aux ressources. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	Chaîne	 Oui	 Oui	 Oui
condition .aws : Principal OrgPaths	ID de l'organisation ou de l'unité organisationnelle (UO) indiqué comme condition d'accès aux ressources. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	Chaîne	 Oui	 Oui	 Oui
condition .aws : SourceIp	Adresse IP qui autorise l'accès du principal à la ressource lors de l'utilisation de l'adresse IP spécifiée. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	Adresse IP	 Oui	 Oui	 Oui

Criterion	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès
condition .aws : SourceVpc	ID de VPC qui autorise l'accès du principal à la ressource lors de l'utilisation du VPC spécifié. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	Chaîne	 Oui	 Oui	 Oui
condition .aws : UserId	ID de l'utilisateur IAM à partir d'un compte externe indiqué comme condition d'accès à la ressource. Pour en savoir plus, consultez Clés de contexte de condition globale AWS.	Chaîne	 Oui	 Oui	 Oui
condition .cognito-identity. amazonaws .com:aud	ID de pool d'identités Amazon Cognito spécifié comme condition d'accès au rôle IAM dans le résultat. Pour en savoir plus, consultez les rubriques IAM et clés AWS STS contextuelles de condition.	Chaîne	 Oui	 Oui	 Oui

Criterion	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès
condition.graph.facebook.com:app_id	ID d'application Facebook (ou ID de site) spécifié comme condition permettant d'autoriser l'accès de la fédération Login with Facebook au rôle IAM dans le résultat. Pour en savoir plus, consultez les rubriques IAM et clés AWS STS contextuelles de condition.	Chaîne	 Oui	 Oui	 Oui
condition.accounts.google.com:aud	ID d'application Google spécifié comme condition d'accès au rôle IAM. Pour en savoir plus, consultez les rubriques IAM et clés AWS STS contextuelles de condition.	Chaîne	 Oui	 Oui	 Oui

Criterion	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès
condition . km : CallerAccount	ID de AWS compte propriétaire de l'entité appelante (utilisateur IAM, rôle ou utilisateur root du compte) utilisée par les services qui appellent AWS KMS. Pour en savoir plus, consultez la section Clés de condition pour AWS Key Management Service.	Chaîne	 Oui	 Oui	 Oui
condition .www.amazon.com:app_id	ID d'application Amazon (ou ID de site) spécifié comme condition permettant d'autoriser l'accès de la fédération Login with Amazon au rôle. Pour en savoir plus, veuillez consulter	Chaîne	 Oui	 Oui	 Oui
id	ID du résultat.	Chaîne	 Non	 Oui	 Oui

Criterion	Description	Type	Règle d'archivage	Liste des résultats	Liste des résultats de la prévisualisation de l'accès
changeType	Donne un contexte sur la comparaison du résultat de la prévisualisation de l'accès à celui existant et identifié dans l'IAM Access Analyzer.	Chaîne	 Non	 Non	 Oui
existingFindingId	L'ID existant du résultat dans l'IAM Access Analyzer, fourni uniquement pour les résultats trouvés dans la prévisualisation de l'accès.	Chaîne	 Non	 Non	 Oui
existingFindingStatus	Le statut existant du résultat, fourni uniquement pour les résultats existants dans la prévisualisation de l'accès.	Chaîne	 Non	 Non	 Oui

Utilisation des rôles liés aux services pour AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer utilise un rôle lié à un service IAM. Un rôle lié à un service est un type unique de rôle IAM lié directement à l'analyseur d'accès IAM. Les rôles liés aux services sont prédéfinis par IAM Access Analyzer et incluent toutes les autorisations requises par la fonctionnalité pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service simplifie la configuration de l'analyseur d'accès IAM, car il n'est pas nécessaire d'ajouter manuellement les autorisations nécessaires. L'analyseur d'accès IAM définit les autorisations des rôles liés à ses services et, sauf indication contraire, seul l'analyseur d'accès IAM peut assumer ces rôles. Les autorisations définies comprennent la politique d'approbation et la politique d'autorisation. De plus, cette politique d'autorisation ne peut pas être attachée à une autre entité IAM.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés aux services, consultez Services AWS fonctionnant avec IAM et recherchez les services où Oui figure dans la colonne Rôle lié à un service. Sélectionnez un Oui ayant un lien pour consulter la documentation du rôle lié à un service, pour ce service.

Autorisations des rôles liés à un service pour AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer utilise le rôle lié au service nommé AWSServiceRoleForAccessAnalyzer— Allow Access Analyzer pour analyser les métadonnées des ressources pour un accès externe et pour analyser l'activité afin d'identifier les accès non utilisés.

Le rôle AWSServiceRoleForAccessAnalyzer lié à un service fait confiance aux services suivants pour assumer le rôle :

- `access-analyzer.amazonaws.com`

La politique d'autorisations de rôle nommée `AccessAnalyzerServiceRolePolicy` autorise l'analyseur d'accès IAM à effectuer des actions sur des ressources spécifiques.

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour plus d'informations, consultez Service-Linked Role Permissions (autorisations du rôle lié à un service) dans le IAM User Guide (guide de l'utilisateur IAM).

Création d'un rôle lié à un service pour l'analyseur d'accès IAM

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous activez Access Analyzer dans l'API AWS Management Console ou dans l' AWS API, IAM Access Analyzer crée le rôle lié au service pour vous. Le même rôle lié à un service est utilisé dans toutes les régions dans lesquelles vous activez IAM Access Analyzer. Les résultats des accès externes et non utilisés utilisent le même rôle lié au service.

> **Note**
>
> L’IAM Access Analyzer est régional. Vous devez activer l’IAM Access Analyzer dans chaque Région de manière indépendante.

Si vous supprimez ce rôle lié à un service, IAM Access Analyzer recrée le rôle lors de la prochaine création d'un analyseur.

Vous pouvez également utiliser la console IAM pour créer un rôle lié au service avec le cas d'utilisation Access Analyzer. Dans l'API AWS CLI ou dans l' AWS API, créez un rôle lié à un service avec le nom du `access-analyzer.amazonaws.com` service. Pour plus d'informations, consultez Création d'un rôle lié à un service dans le Guide de l'utilisateur IAM. Si vous supprimez ce rôle lié à un service, vous pouvez utiliser ce même processus pour créer le rôle à nouveau.

Modification d’un rôle lié à un service pour l’analyseur d’accès IAM

IAM Access Analyzer ne vous permet pas de modifier le rôle lié au AWSServiceRoleForAccessAnalyzer service. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l’aide d’IAM. Pour plus d’informations, consultez Modification d’un rôle lié à un service dans le Guide de l’utilisateur IAM.

Suppression d’un rôle lié à un service pour l’analyseur d’accès IAM

Si vous n’avez plus besoin d’utiliser une fonctionnalité ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, aucune entité inutilisée n'est surveillée ou gérée activement. Cependant, vous devez nettoyer les ressources de votre rôle lié à un service avant de pouvoir les supprimer manuellement.

> **Note**
>
> Si l’analyseur d’accès IAM utilise le rôle lorsque vous essayez de supprimer les ressources, alors la suppression peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour supprimer les ressources IAM Access Analyzer utilisées par AWSServiceRoleForAccessAnalyzer

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans la section Access reports (Rapports d'accès) sous Access Analyzer (Analyseur d'accès), sélectionnez Analyzers (Analyseurs).
3. Cochez la case située en haut à gauche au-dessus de la liste des analyseurs dans le tableau Analyzers (Analyseurs) pour sélectionner tous les analyseurs.
4. Sélectionnez Delete (Supprimer).
5. Pour confirmer que vous souhaitez supprimer les analyseurs, entrez **`delete`**, puis sélectionnez Delete (Supprimer).

Pour supprimer manuellement le rôle lié à un service à l'aide d'IAM

Utilisez la console IAM, le AWS CLI, ou l' AWS API pour supprimer le rôle lié au AWSServiceRoleForAccessAnalyzer service. Pour plus d'informations, consultez Suppression d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Régions prises en charge pour les rôles liés à un service de l'analyseur d'accès IAM

L'analyseur d'accès IAM prend en charge l'utilisation des rôles liés à un service dans toutes les régions où le service est disponible. Pour plus d'informations, consultez Régions et Points de terminaison AWS.

Prévisualiser l'accès

En plus de vous aider à identifier les ressources partagées avec une entité externe, AWS IAM Access Analyzer vous donne également un aperçu des résultats d'IAM Access Analyzer avant de déployer les autorisations des ressources afin que vous puissiez vérifier que vos modifications de politique accordent uniquement l'accès public et multicompte prévu à votre ressource. Vous pouvez ainsi commencer avec l'accès externe prévu à vos ressources.

Vous pouvez prévisualiser et vérifier l'accès public et intercompte à vos compartiments Amazon S3 dans la console Amazon S3. Vous pouvez également utiliser les API IAM Access Analyzer pour prévisualiser l'accès public et multicompte à vos compartiments Amazon S3, à vos AWS KMS clés, à vos rôles IAM, à vos files d'attente Amazon SQS et aux secrets de Secrets Manager en proposant des autorisations pour votre ressource.

Rubriques

- Prévisualiser l'accès dans la console Amazon S3
- Prévisualiser l'accès avec les API de l’IAM Access Analyzer

Prévisualiser l'accès dans la console Amazon S3

Une fois votre politique de compartiment réalisée dans la console Amazon S3, vous avez la possibilité de prévisualiser l'accès public et intercompte à votre compartiment Amazon S3. Vous pouvez vérifier que vos modifications de politique octroient uniquement l'accès externe prévu avant de choisir Save changes (Enregistrer les modifications). Cette étape facultative vous permet de prévisualiser AWS Identity and Access Management Access Analyzer les résultats de votre bucket. Vous pouvez vérifier si la modification de politique introduit de nouveaux résultats ou résout les résultats existants pour l'accès externe. Vous pouvez ignorer cette étape de validation et enregistrer votre politique de compartiment Amazon S3 à tout moment.

Pour prévisualiser l'accès externe à votre compartiment, vous devez disposer d'un analyseur de compte actif dans la région de votre compartiment avec le compte comme zone de confiance. Vous devez également avoir des autorisations requises pour l’utilisation de l’IAM Access Analyzer et sa prévisualisation Pour plus d'informations sur l'activation de l’IAM Access Analyzer et les autorisations requises, veuillez consulter Activation de l’IAM Access Analyzer.

Pour prévisualiser l'accès à votre compartiment Amazon S3 lorsque vous créez ou modifiez votre politique de compartiment

1. Une fois la création ou la modification de votre politique de compartiment terminée, assurez-vous que votre politique est une politique de compartiment Amazon S3 valide. L'ARN de politique doit correspondre à l'ARN du compartiment, et les éléments de politique doivent être valides.
2. En dessous de la politique, sous Preview external access (Prévisualiser l'accès externe), sélectionnez un analyseur de comptes actif, puis sélectionnez Preview (Prévisualiser). Une prévisualisation des résultats d’IAM Access Analyzer est générée pour votre compartiment. La prévisualisation analyse la politique de compartiment Amazon S3 affichée, ainsi que les autorisations de compartiment existantes. Cela inclut les paramètres BPA du compartiment et du compte, l'ACL du compartiment, les points d'accès Amazon S3 et les points d'accès multi-région attachés au compartiment, ainsi que leurs politiques et paramètres BPA.
3. Une fois la prévisualisation de l'accès terminée, une prévisualisation des résultats d'IAM Access Analyzer s'affiche. Chaque résultat signale une instance d'un principal en dehors du compte avec accès à votre compartiment après avoir enregistré la politique. Vous pouvez valider l'accès

à votre compartiment en analysant chaque résultat. L'en-tête du résultat fournit un résumé de l'accès, et vous pouvez développer le résultat afin d'analyser les détails d'un résultat. La recherche de badges fournit un contexte sur la façon dont l'enregistrement de la politique de compartiment modifierait l'accès au compartiment. Par exemple, ils vous aident à vérifier si la modification de politique introduit de nouveaux résultats ou résout les résultats existants pour un accès externe :

a. Nouveau : indique un résultat pour un nouvel accès externe que la politique introduirait.

b. Résolu : indique un résultat pour un accès externe existant que la politique supprimerait.

c. Archivé : indique un résultat pour un nouvel accès externe qui serait automatiquement archivé en fonction des règles d'archivage de l'analyseur qui définissent à quel moment les résultats doivent être marqués comme prévus.

d. Existant : indique un résultat existant pour un accès externe qui resterait inchangé.

e. Public : si un résultat concerne l'accès public à la ressource, il aura un badge Public en plus de l'un des badges ci-dessus.

4. Si vous identifiez un accès externe que vous n'envisagez pas d'introduire ou de supprimer, vous pouvez réviser la politique, puis choisir Preview (Prévisualiser) jusqu'à ce que vous réalisiez l'accès externe prévu. Si vous avez un résultat étiqueté Public, nous vous recommandons de réviser la politique afin de supprimer l'accès public avant de choisir Save changes (Enregistrer les modifications). La prévisualisation de l'accès est une étape facultative, et vous pouvez choisir Save changes (Enregistrer les modifications) à tout moment.

Prévisualiser l'accès avec les API de l’IAM Access Analyzer

Vous pouvez utiliser les API IAM Access Analyzer pour prévisualiser l'accès public et entre comptes à vos compartiments Amazon S3, à vos AWS KMS clés, à vos rôles IAM, à vos files d'attente Amazon SQS et à vos secrets Secrets Manager. Vous pouvez prévisualiser l'accès en fournissant des autorisations proposées pour une ressource existante que vous possédez ou une nouvelle ressource que vous voulez déployer.

Pour prévisualiser l'accès externe à votre ressource, vous devez disposer d'un analyseur de compte actif pour le compte et la région de la ressource. Vous devez également avoir des autorisations requises pour l’utilisation de l’IAM Access Analyzer et sa prévisualisation Pour plus d'informations sur l'activation de l’IAM Access Analyzer et les autorisations requises, veuillez consulter Activation de l’IAM Access Analyzer.

Pour prévisualiser l'accès à une ressource, vous pouvez utiliser l'opération `CreateAccessPreview` et fournir l'ARN de l'analyseur et la configuration du contrôle d'accès pour la ressource. Le service renvoie l'ID unique pour la prévisualisation de l'accès, que vous pouvez utiliser pour en vérifier l'état avec l'opération `GetAccessPreview`. Lorsque le statut est `Completed`, vous pouvez utiliser l'opération `ListAccessPreviewFindings` pour récupérer les résultats générés pour la prévisualisation de l'accès. Les opérations `GetAccessPreview` et `ListAccessPreviewFindings` récupèrent les prévisualisations d'accès et les résultats créés dans un délai d'environ 24 heures.

Chaque résultat récupéré contient des détails du résultat, qui décrivent l'accès. Un statut de prévisualisation du résultat indiquant si le résultat serait `Active`, `Archived` ou `Resolved` après le déploiement des autorisations, et un `changeType`. Le `changeType` donne un contexte sur la comparaison du résultat de la prévisualisation de l'accès à celui existant et identifié dans l'IAM Access Analyzer :

- Nouveau : le résultat concerne l'accès nouvellement introduit.
- Inchangé : le résultat de la prévisualisation est un résultat existant qui resterait inchangé.
- Modifié : le résultat de la prévisualisation est un résultat existant dont le statut changerait.

Le `status` et le `changeType` vous aident à comprendre comment la configuration de la ressource modifierait l'accès à la ressource existante. Si le `changeType` est `Unchanged` ou Modifié, le résultat contiendra également l'ID et le statut existants du résultat dans l'IAM Access Analyzer. Par exemple, un résultat `Changed` avec le statut de prévisualisation `Resolved` et un statut existant `Active` indique que le résultat existant `Active` pour la ressource deviendrait `Resolved` à la suite de la modification proposée des autorisations.

Vous pouvez utiliser l'opération `ListAccessPreviews` pour récupérer une liste des prévisualisations d'accès pour l'analyseur spécifié. Cette opération récupère les informations sur la prévisualisation de l'accès créée dans un délai d'environ une heure.

En général, si la prévisualisation de l'accès concerne une ressource existante et que vous ne spécifiez pas d'option de configuration, la prévisualisation de l'accès utilisera la configuration de ressource existante par défaut. Si la prévisualisation de l'accès concerne une nouvelle ressource et que vous ne spécifiez pas d'option de configuration, la prévisualisation de l'accès utilisera la valeur par défaut en fonction du type de ressource. Pour connaître les cas de configuration pour chaque type de ressource, veuillez consulter ce qui suit.

Prévisualiser l'accès à votre compartiment Amazon S3

Pour créer une prévisualisation de l'accès pour un nouveau compartiment Amazon S3 ou un compartiment Amazon S3 existant que vous possédez, vous pouvez proposer une configuration de compartiment en spécifiant la politique de compartiment Amazon S3, les ACL de compartiment, les paramètres BPA de compartiment et les points d'accès Amazon S3, y compris les points d'accès multi-région, attachés au compartiment.

> **Note**
>
> Avant de tenter de créer un aperçu d'accès pour un nouveau compartiment, nous vous recommandons d'appeler l'HeadBucketopération Amazon S3 pour vérifier si le compartiment nommé existe déjà. Cette opération est utile pour déterminer si un compartiment existe et si vous avez l'autorisation d'y accéder.

Politique de compartiment : si la configuration concerne un compartiment Amazon S3 existant et que vous ne spécifiez pas la politique de compartiment Amazon S3, la prévisualisation de l'accès utilise la politique existante attachée au compartiment. Si la prévisualisation de l'accès concerne une nouvelle ressource et que vous ne spécifiez pas la politique de compartiment Amazon S3, la prévisualisation de l'accès suppose un compartiment sans politique. Pour proposer la suppression d'une politique de compartiment existante, vous pouvez spécifier une chaîne vide. Pour plus d'informations sur les limites de la politique de compartiment prise en charge, veuillez consulter Bucket policy examples (Exemples de politique de compartiment).

Octrois d'ACL de compartiment : vous pouvez proposer jusqu'à 100 octrois d'ACL par compartiment. Si la configuration d'octroi proposée concerne un compartiment existant, la prévisualisation de l'accès utilise la liste proposée des configurations d'octroi à la place des autorisations existantes. Sinon, la prévisualisation de l'accès utilise les autorisations existantes pour le compartiment.

Points d'accès au compartiment : l'analyse prend en charge jusqu'à 100 points d'accès, dont des points d'accès multi-région, par compartiment, et vous pouvez proposer jusqu'à dix nouveaux points d'accès par compartiment. Si la configuration de point d'accès Amazon S3 proposée concerne un compartiment existant, la prévisualisation de l'accès utilise la configuration proposée de points d'accès à la place des points d'accès existants. Pour proposer un point d'accès sans politique, vous pouvez fournir une chaîne vide en tant que politique de point d'accès. Pour plus d'informations sur les limites de la politique de point d'accès, veuillez consulter Access points restrictions and limitations (Limites et restrictions des points d'accès).

Bloquer la configuration de l'accès public : si la configuration proposée concerne un compartiment Amazon S3 existant et que vous ne spécifiez pas la configuration, la prévisualisation de l'accès utilise le paramètre existant. Si la configuration proposée concerne un nouveau compartiment et que vous ne spécifiez pas la configuration BPA du compartiment, la prévisualisation de l'accès utilise `false`. Si la configuration proposée concerne un nouveau point d'accès ou point d'accès multi-région et que vous ne spécifiez pas la configuration BPA du point d'accès, la prévisualisation de l'accès utilise `true`.

Prévisualiser l'accès à votre clé AWS KMS

Pour créer un aperçu de l'accès pour une nouvelle AWS KMS clé ou une AWS KMS clé existante que vous possédez, vous pouvez proposer une configuration de AWS KMS clé en spécifiant la politique des clés et la configuration des AWS KMS autorisations.

AWS KMS politique de clé — Si la configuration concerne une clé existante et que vous ne spécifiez pas la politique de clé, l'aperçu de l'accès utilise la politique existante pour la clé. Si la prévisualisation de l'accès concerne une nouvelle ressource et que vous ne spécifiez pas la politique de clé, la prévisualisation de l'accès utilise la politique de clé par défaut. La politique de clé proposée ne peut pas être une chaîne vide.

AWS KMS subventions — L'analyse prend en charge jusqu'à 100 autorisations KMS par configuration*.* Si la configuration de subvention proposée concerne une clé existante, l'aperçu de l'accès utilise la liste proposée de configurations de subventions à la place des subventions existantes. Sinon, la prévisualisation de l'accès utilise les octrois existants pour la clé.

Prévisualiser l'accès à votre rôle IAM

Pour créer une prévisualisation de l'accès pour un nouveau rôle IAM ou un rôle IAM existant que vous possédez, vous pouvez proposer une configuration de rôle IAM en spécifiant la politique d'approbation.

Politique d'approbation de rôle : si la configuration concerne un nouveau rôle IAM, vous devez spécifier la politique d'approbation. Si la configuration concerne un rôle IAM existant que vous possédez et que vous ne proposez pas la politique d'approbation, la prévisualisation de l'accès utilise la politique d'approbation existante pour le rôle. La politique d'approbation proposée ne peut pas être une chaîne vide.

Prévisualiser l'accès à votre file d'attente Amazon SQS

Pour créer une prévisualisation de l'accès pour une nouvelle file d'attente Amazon SQS ou une file d'attente Amazon SQS existante que vous possédez, vous pouvez proposer une configuration de file d'attente Amazon SQS en spécifiant la politique Amazon SQS pour la file d'attente.

Politique de file d'attente Amazon SQS : si la configuration concerne une file d'attente Amazon SQS existante et que vous ne spécifiez pas la politique Amazon SQS, la prévisualisation de l'accès utilise la politique Amazon SQS existante pour la file d'attente. Si la prévisualisation de l'accès concerne une nouvelle ressource et que vous ne spécifiez pas la politique, la prévisualisation de l'accès suppose une file d'attente Amazon SQS sans politique. Pour proposer la suppression d'une politique de file d'attente Amazon SQS existante, vous pouvez spécifier une chaîne vide pour la politique Amazon SQS.

Prévisualiser l'accès à votre secret Secrets Manager

Pour créer un aperçu de l'accès à un nouveau secret du Gestionnaire de Secrets ou à un secret du Gestionnaire de Secrets existant dont vous êtes propriétaire, vous pouvez proposer une configuration du secret du Gestionnaire de secrets en spécifiant la politique secrète et la clé de AWS KMS chiffrement optionnelle.

Politique de secret : si la configuration concerne un secret existant et que vous ne spécifiez pas la politique secrète, la prévisualisation de l'accès utilise la politique existante pour le secret. Si la prévisualisation de l'accès concerne une nouvelle ressource et que vous ne spécifiez pas la politique, la prévisualisation de l'accès suppose un secret sans politique. Pour proposer la suppression d'une politique existante, vous pouvez spécifier une chaîne vide.

AWS KMS clé de chiffrement — Si la configuration proposée concerne un nouveau secret et que vous ne spécifiez pas l'ID de la AWS KMS clé, l'aperçu de l'accès utilise la clé KMS par défaut du AWS compte. Si vous spécifiez une chaîne vide pour l'ID de AWS KMS clé, l'aperçu de l'accès utilise la clé KMS par défaut du AWS compte.

Vérifications de validation des politiques

L’analyseur d’accès IAM fournit des contrôles de politique qui aident à valider vos politiques IAM avant de les attacher à une entité. Il s’agit notamment des vérifications de politique de base fournies par la validation des politiques pour valider votre politique par rapport à la grammaire des politiques et aux bonnes pratiques AWS. Vous pouvez afficher les résultats des vérifications de validation de

politique qui incluent des avertissements de sécurité, des erreurs, des avertissements généraux et des suggestions pour votre politique.

Vous pouvez utiliser des vérifications de politique personnalisées pour vérifier les nouveaux accès en fonction de vos normes de sécurité. Des frais sont associés à chaque vérification pour un nouvel accès. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

Rubriques

- Validation de la politique de l'IAM Access Analyzer
- Vérifications de politiques personnalisées de l'analyseur d'accès IAM

Validation de la politique de l'IAM Access Analyzer

Vous pouvez valider vos politiques à l'aide de AWS Identity and Access Management Access Analyzer la validation des politiques. Vous pouvez créer ou modifier une politique à l'aide de l' AWS CLI AWS API ou de l'éditeur de stratégie JSON dans la console IAM. L'analyseur d'accès IAM valide votre politique par rapport à la grammaire de politique IAM et aux bonnes pratiques AWS. Vous pouvez afficher les résultats des vérifications de validation de politique qui incluent des avertissements de sécurité, des erreurs, des avertissements généraux et des suggestions pour votre politique. Ces résultats fournissent des recommandations exploitables qui vous aident à créer des politiques fonctionnelles et conformes aux bonnes pratiques en matière de sécurité. Pour afficher la liste des vérifications des politiques de base exécutées par l'analyseur d'accès IAM, consultez Référence de vérification de politique d'analyseur d'accès.

Validation des politiques dans IAM (console)

Vous pouvez afficher les résultats générés par la validation de politiques de l'analyseur d'accès IAM lorsque vous créez ou modifiez une politique gérée dans la console IAM. Vous pouvez également afficher ces résultats pour les politiques d'utilisateur ou de rôle en ligne. L'analyseur d'accès IAM ne génère pas ces résultats pour les politiques de groupe en ligne.

Pour afficher les résultats générés par les vérifications de politique pour les politiques IAM JSON

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Commencez à créer ou à modifier une politique selon l'une des méthodes suivantes :

a. Pour créer une politique gérée, accédez à la page Policies (Politiques) et créez une politique. Pour plus d'informations, consultez Création de politiques à l'aide de l'éditeur JSON.

b. Pour afficher les vérifications de politique d'une politique gérée par le client existante, accédez à la page Politiques, sélectionnez le nom d'une politique, puis sélectionnez Modifier. Pour plus d'informations, consultez Modification de politiques gérées par le client (console).

c. Pour afficher les vérifications de politique d'une politique en ligne sur un utilisateur ou un rôle, accédez à la page Utilisateurs ou Rôles, sélectionnez le nom d'un utilisateur ou d'un rôle, sélectionnez le nom de la politique sous l'onglet Autorisations, puis choisissez Modifier. Pour plus d'informations, consultez Modification de politiques gérées par le client (console).

3. Dans l'éditeur de politique, sélectionnez l'onglet JSON.

4. Dans le panneau de validation de la politique, sous la politique, sélectionnez un ou plusieurs des onglets suivants. Les noms des onglets indiquent également le nombre de chaque type de résultat pour votre politique.

   - Sécurité : consultez les avertissements si votre politique autorise un accès AWS considéré comme présentant un risque de sécurité en raison d'un accès trop permissif.
   - Erreurs : affiche des erreurs si votre politique inclut des lignes qui empêchent la politique de fonctionner.
   - Avertissements : affiche des avertissements si votre politique n'est pas conforme aux bonnes pratiques, mais que les problèmes ne constituent pas des risques de sécurité.
   - Suggestions : affiche des suggestions si AWS recommande des améliorations qui n'affectent pas les autorisations de la politique.

5. Examinez les détails des résultats fournis par la vérification de la politique de l'IAM Access Analyzer. Chaque résultat indique l'emplacement du problème signalé. Pour en savoir plus sur les causes du problème et la façon de le résoudre, sélectionnez le lien Learn more (En savoir plus) en regard du résultat. Vous pouvez également rechercher la vérification de politique associée à chaque résultat sur la page de référence Access Analyzer policy checks (Vérifications de politique Access Analyzer).

6. Facultatif. Si vous modifiez une politique existante, vous pouvez exécuter une vérification de politique personnalisée pour déterminer si votre politique mise à jour accorde un nouvel accès par rapport à la version existante. Dans le panneau de validation de la politique ci-dessous, sélectionnez l'onglet Vérifier un nouvel accès, puis sélectionnez Vérifier la politique. Si les

autorisations modifiées accordent un nouvel accès, la déclaration sera mise en évidence dans le volet de validation de la politique. Si vous n'avez pas l'intention d'accorder un nouvel accès, mettez à jour la déclaration de politique et choisissez Vérifier la politique jusqu'à ce qu'aucun nouvel accès ne soit détecté. Pour plus d'informations, consultez Vérifications de politiques personnalisées de l'analyseur d'accès IAM.

> **Note**
>
> Des frais sont associés à chaque vérification pour un nouvel accès. Pour plus d'informations sur la tarification, consultez Tarification de l'analyseur d'accès IAM.

7. Mettez à jour votre politique pour résoudre les résultats.

> **Important**
>
> Testez soigneusement les politiques nouvelles ou modifiées avant de les implémenter dans votre flux de production.

8. Lorsque vous avez terminé, choisissez Next. Le Validateur de politique signale les erreurs de syntaxe qui ne sont pas signalées par l'analyseur d'accès IAM.

> **Note**
>
> Vous pouvez basculer à tout moment entre les onglets Visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'onglet Visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez Restructuration de politique.

9. Pour de nouvelles politiques, sur la page Examiner et créer, tapez un Nom de politique et une Description (facultative) pour la politique que vous créez. Examinez les Autorisations définies dans cette politique pour voir celles accordées par votre politique. Sélectionnez ensuite Créer une politique pour enregistrer votre travail.

   Pour les politiques existantes, à la page Examiner et enregistrer, passez en revue les Autorisations définies dans cette politique pour voir celles qui sont accordées par votre politique. Choisissez l'option Définir cette nouvelle version comme case à cocher par défaut pour enregistrer la version mise à jour comme version par défaut de la politique. Choisissez ensuite Enregistrer les modifications pour enregister votre travail.

Validation des politiques à l'aide d'IAM Access Analyzer (ou API)AWS CLIAWS

Vous pouvez afficher les résultats générés par la validation de politique de l’analyseur d’accès IAM à partir d’ AWS Command Line Interface (AWS CLI).

Pour consulter les résultats générés par la validation des politiques (AWS CLI ou AWS API) d'IAM Access Analyzer

Utilisez l'une des options suivantes :

- AWS CLI : aws accessanalyzer validate-policy
- AWS API : ValidatePolicy

Référence de vérification de politique d'analyseur d'accès

Vous pouvez valider vos politiques à l'aide de AWS Identity and Access Management Access Analyzer la validation des politiques. Vous pouvez créer ou modifier une politique à l'aide de l' AWS CLI AWS API ou de l'éditeur de stratégie JSON dans la console IAM. L’analyseur d’accès IAM valide votre politique par rapport à la grammaire de politique IAM et aux bonnes pratiques AWS. Vous pouvez afficher les résultats des vérifications de validation de politique qui incluent des avertissements de sécurité, des erreurs, des avertissements généraux et des suggestions pour votre politique. Ces résultats fournissent des recommandations exploitables qui vous aident à créer des politiques fonctionnelles et conformes aux bonnes pratiques en matière de sécurité. La liste des vérifications de politiques de base fournies par l’analyseur d’accès IAM est présentée ci-dessous. Pas de frais supplémentaires pour l’exécution des vérifications de validation de la politique. Pour en savoir plus sur la validation des politiques à l’aide de la validation des politiques, consultez Validation de la politique de l’IAM Access Analyzer.

Erreur : compte d'ARN non autorisé

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
ARN account not allowed: The service {{service}} does not support specifying an account
 ID in the resource ARN.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The service {{service}} does not support specifying an account ID in
 the resource ARN."
```

Résolution de l'erreur

Supprimez l'ID de compte de l'ARN de ressource. Les ARN des ressources pour certains AWS services ne permettent pas de spécifier un identifiant de compte.

Par exemple, Amazon S3 ne prend pas en charge un ID de compte en tant qu'espace de noms dans les ARN de compartiment. Le nom d'un compartiment Amazon S3 est unique au monde et l'espace de noms est partagé par tous les AWS comptes. Pour afficher tous les types de ressources disponibles dans Amazon S3, veuillez consulter Types de ressources définis par Amazon S3 dans la Référence des autorisations de service.

Termes connexes

- Ressources de politique
- Identificateurs de compte
- ARN de la ressource
- AWS ressources de service avec des formats ARN

Erreur : région d'ARN non autorisée

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
ARN Region not allowed: The service {{service}} does not support specifying a Region in
 the resource ARN.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The service {{service}} does not support specifying a Region in the
 resource ARN."
```

Résolution de l'erreur

Supprimez la région de l'ARN de ressource. Les ARN des ressources pour certains AWS services ne permettent pas de spécifier une région.

Par exemple, IAM est un service global. La partie Région d'un ARN de ressource IAM est toujours vide. Les ressources IAM sont mondiales, comme un AWS compte l'est aujourd'hui. Par exemple, une fois connecté en tant qu'utilisateur IAM, vous pouvez accéder aux AWS services dans n'importe quelle région géographique.

- Ressources de politique
- ARN de la ressource
- AWS ressources de service avec des formats ARN

Erreur : non-correspondance du type de données

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Data type mismatch: The text does not match the expected JSON data type {{data_type}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The text does not match the expected JSON data type {{data_type}}."
```

Résolution de l'erreur

Mettez à jour le texte pour utiliser le type de données pris en charge.

Par exemple, la clé de condition globale `Version` a besoin d'un type de données `String`. Si vous fournissez une date ou un entier, le type de données ne correspondra pas.

Termes connexes

- Clés de condition globale
- Éléments de politique JSON IAM : Opérateurs de condition

Erreur : clés dupliquées avec une casse différente

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Duplicate keys with different case: The condition key {{key}} appears more than once
 with different capitalization in the same condition block. Remove the duplicate
 condition keys.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key {{key}} appears more than once with different
 capitalization in the same condition block. Remove the duplicate condition keys."
```

Résolution de l'erreur

Examinez les clés de condition similaires dans le même bloc de condition et utilisez la même casse pour toutes les instances.

On appelle bloc de condition le texte inclus dans l'élément `Condition`d'une instruction de politique. Les noms de clé de condition ne sont pas sensibles à la casse. La sensibilité à la casse des valeurs des clés de condition dépend de l'opérateur de condition que vous utilisez. Pour de plus amples informations sur la sensibilité à la casse des clés de condition, veuillez consulter Éléments de politique JSON IAM : Condition.

Termes connexes

- Conditions
- Bloc de condition
- Clés de condition globale
- AWS clés d'état de service

Erreur : action non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid action: The action {{action}} does not exist. Did you mean {{valid_action}}?
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The action {{action}} does not exist. Did you mean
 {{valid_action}}?"
```

Résolution de l'erreur

L'action que vous avez spécifiée n'est pas valide. Cela peut se produire si vous saisissez de façon incorrecte le préfixe du service ou le nom de l'action. Pour certains problèmes courants, la vérification de politique renvoie une action suggérée.

Termes connexes

- Actions de politique
- AWS actions de service

AWS politiques gérées avec cette erreur

AWS les politiques gérées vous permettent de démarrer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Les politiques AWS gérées suivantes incluent des actions non valides dans leurs déclarations de politique. Les actions non valides n'affectent pas les autorisations octroyées par la politique. Lorsque vous utilisez une stratégie AWS gérée comme référence pour créer votre stratégie gérée, il est AWS recommandé de supprimer les actions non valides de votre stratégie.

- Amazon EMR _v2 FullAccessPolicy
- CloudWatchSyntheticsFullAccess

Erreur : compte d'ARN non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid ARN account: The resource ARN account ID {{account}} is not valid. Provide a
 12-digit account ID.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The resource ARN account ID {{account}} is not valid. Provide a 12-
digit account ID."
```

Résolution de l'erreur

Mettez à jour l'ID de compte dans l'ARN de ressource. Les ID de compte sont des nombres entiers à 12 chiffres. Pour savoir comment consulter l'identifiant de votre compte, consultez la section Trouver votre identifiant de AWS compte.

Termes connexes

- Ressources de politique
- Identificateurs de compte
- ARN de la ressource
- AWS ressources de service avec des formats ARN

Erreur : préfixe d'ARN non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid ARN prefix: Add the required prefix (arn) to the resource ARN.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add the required prefix (arn) to the resource ARN."
```

Résolution de l'erreur

AWS Les ARN des ressources doivent inclure le `arn:` préfixe requis.

Termes connexes

- Ressources de politique
- ARN de la ressource
- AWS ressources de service avec des formats ARN

Erreur : région d'ARN non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid ARN Region: The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region."
```

Résolution de l'erreur

Le type de ressource n'est pas pris en charge dans la région spécifiée. Pour un tableau des AWS services pris en charge dans chaque région, consultez le tableau des régions.

Termes connexes

- Ressources de politique
- ARN de la ressource
- Noms et codes des régions

Erreur : ARN de ressource non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid ARN resource: Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN."
```

Résolution de l'erreur

L'ARN de ressource doit correspondre aux spécifications des types de ressources connus. Pour afficher le format ARN attendu pour un service, voir Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service pour afficher ses types de ressources et formats d'ARN.

Termes connexes

- Ressources de politique

- ARN de la ressource
- AWS ressources de service avec des formats ARN

Erreur : cas de service ARN non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid ARN service case: Update the service name ${service} in the resource ARN to use
 all lowercase letters.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Update the service name ${service} in the resource ARN to use all
 lowercase letters."
```

Résolution de l'erreur

Le service dans l'ARN de ressource doit correspondre aux spécifications (notamment la casse) des préfixes de service. Pour afficher le préfixe d'un service, consultez la section Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service et localisez son préfixe dans la première phrase.

Termes connexes

- Ressources de politique
- ARN de la ressource
- AWS ressources de service avec des formats ARN

Erreur : type de données de condition non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid condition data type: The condition value data types do not match. Use condition
 values of the same JSON data type.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition value data types do not match. Use condition values of
 the same JSON data type."
```

Résolution de l'erreur

La valeur de la paire clé-valeur de condition doit correspondre au type de données de la clé de condition et de l'opérateur de condition. Pour afficher le type de données de clé de condition d'un service, consultez la section Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service pour afficher les clés de condition de ce service.

Par exemple, la clé de condition globale `CurrentTime` prend en charge l'opérateur de condition `Date`. Si vous fournissez une chaîne ou un entier pour la valeur dans le bloc de condition, le type de données ne correspondra pas.

Termes connexes

- Conditions
- Bloc de condition
- Éléments de politique JSON IAM : Opérateurs de condition
- Clés de condition globale
- AWS clés d'état de service

Erreur : format de clé de condition non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid condition key format: The condition key format is not valid. Use the format
 service:keyname.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key format is not valid. Use the format
 service:keyname."
```

Résolution de l'erreur

La clé de la paire clé-valeur de condition doit correspondre aux spécifications du service. Pour afficher les clés de condition d'un service, consultez la section Actions, ressources et clés de

condition pour les AWS services. Choisissez le nom du service pour afficher les clés de condition de ce service.

Termes connexes

- Conditions
- Clés de condition globale
- AWS clés d'état de service

Erreur : valeurs booléennes multiples de condition non valides

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid condition multiple Boolean: The condition key does not support multiple Boolean
 values. Use a single Boolean value.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key does not support multiple Boolean values. Use a
 single Boolean value."
```

Résolution de l'erreur

La clé de la paire clé-valeur de condition attend une valeur booléenne unique. Lorsque vous fournissez plusieurs valeurs booléennes, la correspondance de condition peut ne pas renvoyer les résultats attendus.

Pour afficher les clés de condition d'un service, consultez la section Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service pour afficher les clés de condition de ce service.

- Conditions
- Clés de condition globale
- AWS clés d'état de service

Erreur : opérateur de condition non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid condition operator: The condition operator {{operator}} is not valid. Use a
 valid condition operator.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition operator {{operator}} is not valid. Use a valid
 condition operator."
```

Résolution de l'erreur

Mettez à jour la condition pour utiliser un opérateur de condition pris en charge.

Termes connexes

- Éléments de politique JSON IAM : Opérateurs de condition
- Élément de condition
- Présentation des politiques JSON

Erreur : effet non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid effect: The effect {{effect}} is not valid. Use Allow or Deny.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The effect {{effect}} is not valid. Use Allow or Deny."
```

Résolution de l'erreur

Mettez à jour l'élément `Effect` pour utiliser un effet valide. Les valeurs valides pour `Effect` sont **`Allow`** et **`Deny`**.

Termes connexes

- Élément Effet
- Présentation des politiques JSON

Erreur : clé de condition globale non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid global condition key: The condition key {{key}} does not exist. Use a valid
 condition key.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key {{key}} does not exist. Use a valid condition
 key."
```

Résolution de l'erreur

Mettez à jour la clé de condition dans la paire clé-valeur de condition pour utiliser une clé de condition globale prise en charge.

Les clés de condition globales sont des clés de condition avec un `aws:` préfixe. AWS les services peuvent prendre en charge les clés de condition globales ou fournir des clés spécifiques au service qui incluent leur préfixe de service. Par exemple, les clés de condition IAM incluent le préfixe `iam:`. Pour plus d'informations, consultez Actions, ressources et clés de condition pour les AWS services et choisissez le service dont vous souhaitez afficher les clés.

Termes connexes

- Clés de condition globale

Erreur : partition non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid partition: The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}"
```

Résolution de l'erreur

Mettez à jour l'ARN de ressource de sorte à inclure une partition prise en charge. Si vous avez inclus une partition prise en charge, le service ou la ressource peut ne pas prendre en charge la partition que vous avez incluse.

Une partition est un groupe de AWS régions. Chaque AWS compte est limité à une partition. Dans les régions classiques, utilisez la partition `aws`. Dans les régions de Chine, utilisez `aws-cn`.

Termes connexes

- Amazon Resource Names (ARN) : partitions

Erreur : élément de politique non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid policy element: The policy element {{element}} is not valid.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The policy element {{element}} is not valid."
```

Résolution de l'erreur

Mettez à jour la politique de sorte à inclure uniquement les éléments de politique JSON pris en charge.

Termes connexes

- Éléments de politique JSON

Erreur : format de principal non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid principal format: The Principal element contents are not valid. Specify a key-
value pair in the Principal element.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The Principal element contents are not valid. Specify a key-value
 pair in the Principal element."
```

Résolution de l'erreur

Mettez à jour le format de principal de sorte à utiliser un format de paire clé-valeur pris en charge.

Vous pouvez spécifier un principal dans une politique basée sur la ressource, mais pas dans une politique basée sur l'identité.

Par exemple, pour définir l'accès pour tous les utilisateurs d'un AWS compte, utilisez le principe suivant dans votre politique :

```
"Principal": { "AWS": "123456789012" }
```

Termes connexes

- Éléments de politique JSON : principal
- Politiques basées sur l'identité et politiques basées sur les ressources

Erreur : clé principale non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid principal key: The principal key {{principal-key}} is not valid.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The principal key {{principal-key}} is not valid."
```

Résolution de l'erreur

Mettez à jour la clé dans la paire clé-valeur principale pour utiliser une clé principale prise en charge. Les clés principales suivantes sont prises en charge :

- AWS

- CanonicalUser
- Fédérée
- Service

Termes connexes

- Elément principal

Erreur : région non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid Region: The Region {{region}} is not valid. Update the condition value to a
 suported Region.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The Region {{region}} is not valid. Update the condition value to a
 suported Region."
```

Résolution de l'erreur

Mettez à jour la valeur de la paire clé-valeur de condition pour inclure une région prise en charge. Pour un tableau des AWS services pris en charge dans chaque région, consultez le tableau des régions.

Termes connexes

- Ressources de politique
- ARN de la ressource
- Noms et codes des régions

Erreur : service non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid service: The service {{service}} does not exist. Use a valid service name.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The service {{service}} does not exist. Use a valid service name."
```

Résolution de l'erreur

Le préfixe de service dans la clé d'action ou de condition doit correspondre aux spécifications (notamment la casse) des préfixes de service. Pour afficher le préfixe d'un service, voir Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service et localisez son préfixe dans la première phrase.

Termes connexes

- Services connus et leurs actions, ressources et clés de condition

Erreur : clé de condition de service non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid service condition key: The condition key {{key}} does not exist in the service
 {{service}}. Use a valid condition key.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key {{key}} does not exist in the service {{service}}.
 Use a valid condition key."
```

Résolution de l'erreur

Mettez à jour la clé dans la paire clé-valeur de condition pour utiliser une clé de condition connue pour le service. Les noms de clé de condition globale commencent par le préfixe `aws`. Les services AWS peuvent fournir des clés spécifiques au service qui possèdent le préfixe du service. Pour afficher le préfixe d'un service, voir Actions, ressources et clés de condition pour les AWS services.

Termes connexes

- Clés de condition globale

- Services connus et leurs actions, ressources et clés de condition

Erreur : service non valide en action

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid service in action: The service {{service}} specified in the action does not
 exist. Did you mean {{service2}}?
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The service {{service}} specified in the action does not exist. Did
 you mean {{service2}}?"
```

Résolution de l'erreur

Le préfixe de service dans l'action doit correspondre aux spécifications (notamment la casse) des préfixes de service. Pour afficher le préfixe d'un service, voir Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service et localisez son préfixe dans la première phrase.

Termes connexes

- Élément Action
- Services connus et leurs actions

Erreur : variable non valide pour l'opérateur

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid variable for operator: Policy variables can only be used with String and ARN
 operators.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Policy variables can only be used with String and ARN operators."
```

Résolution de l'erreur

Vous pouvez utiliser des variables de politique dans l'élément `Resource` et les comparaisons de chaîne de l'élément `Condition`. Les conditions prennent en charge des variables lorsque vous utilisez des opérateurs chaîne ou des opérateurs ARN. Les opérateurs de chaîne incluent `StringEquals`, `StringLike` et `StringNotLike`. Les opérateurs ARN incluent `ArnEquals` et `ArnLike`. Vous ne pouvez pas utiliser une variable de politique avec d'autres opérateurs, tels que les opérateurs Numeric, Date, Boolean, Binary, IP Address ou Null operators.

Termes connexes

- Utilisation de variables de politique dans l'élément Condition
- Élément de condition

Erreur : version non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid version: The version ${version} is not valid. Use one of the following
 versions: ${versions}
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The version ${version} is not valid. Use one of the following
 versions: ${versions}"
```

Résolution de l'erreur

L'élément `Version` de stratégie spécifie les règles de syntaxe du langage AWS utilisées pour traiter une politique. Pour utiliser toutes les fonctions de politique disponibles, incluez l'élément `Version` suivant avant l'élément `Statement` dans l'ensemble de vos politiques.

```
"Version": "2012-10-17"
```

Termes connexes

- Élément Version

Erreur : erreur de syntaxe Json

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Json syntax error: Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}."
```

Résolution de l'erreur

Votre politique inclut une erreur de syntaxe. Vérifiez votre syntaxe JSON.

Termes connexes

- Validateur JSON
- Référence des éléments de politique JSON IAM
- Présentation des politiques JSON

Erreur : erreur de syntaxe Json

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Json syntax error: Fix the JSON syntax error.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Fix the JSON syntax error."
```

Résolution de l'erreur

Votre politique inclut une erreur de syntaxe. Vérifiez votre syntaxe JSON.

Termes connexes

- Validateur JSON
- Référence des éléments de politique JSON IAM
- Présentation des politiques JSON

Erreur : action manquante

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing action: Add an Action or NotAction element to the policy statement.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add an Action or NotAction element to the policy statement."
```

Résolution de l'erreur

AWS Les politiques JSON doivent inclure un `NotAction` élément `Action` or.

Termes connexes

- Élément Action
- NotAction élément
- Présentation des politiques JSON

Erreur : champ ARN manquant

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing ARN field: Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource"
```

Résolution de l'erreur

Tous les champs de l'ARN de ressource doivent correspondre aux spécifications d'un type de ressource connu. Pour afficher le format ARN attendu pour un service, voir Actions, ressources et clés de condition pour les AWS services. Choisissez le nom du service pour afficher ses types de ressources et formats d'ARN.

Termes connexes

- Ressources de politique
- ARN de la ressource
- AWS ressources de service avec des formats ARN

Erreur : région ARN manquante

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing ARN Region: Add a Region to the {{service}} resource ARN.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a Region to the {{service}} resource ARN."
```

Résolution de l'erreur

Les ARN des ressources pour la plupart des AWS services nécessitent que vous spécifiiez une région. Pour un tableau des AWS services pris en charge dans chaque région, consultez le tableau des régions.

Termes connexes

- Ressources de politique
- ARN de la ressource
- Noms et codes des régions

Erreur : effet manquant

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing effect: Add an Effect element to the policy statement with a value of Allow or
 Deny.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add an Effect element to the policy statement with a value of Allow
 or Deny."
```

Résolution de l'erreur

AWS Les politiques JSON doivent inclure un `Effect` élément dont la valeur est **`Allow`** et**`Deny`**.

Termes connexes

- Élément Effet
- Présentation des politiques JSON

Erreur : principal manquant

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing principal: Add a Principal element to the policy statement.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a Principal element to the policy statement."
```

Résolution de l'erreur

Les politiques basées sur les ressources doivent inclure un élément `Principal`.

Par exemple, pour définir l'accès pour tous les utilisateurs d'un AWS compte, utilisez le principe suivant dans votre politique :

```
"Principal": { "AWS": "123456789012" }
```

Termes connexes

- Elément principal
- Politiques basées sur l'identité et politiques basées sur les ressources

Erreur : qualificateur manquant

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing qualifier: The request context key ${key} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The request context key ${key} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy."
```

Résolution de l'erreur

Dans l'élément `Condition`, vous créez des expressions dans lesquelles vous utilisez des opérateurs de condition tels que égal ou inférieur à pour comparer une condition de la politique avec des clés et des valeurs dans le contexte de la demande. Pour les demandes qui incluent plusieurs valeurs pour une seule clé de condition, vous devez placer les conditions entre crochets comme un tableau ("Key2":["Value2A", "Value2B"]). Vous devez également utiliser les opérateurs définis `ForAllValues` ou `ForAnyValue` avec l'opérateur de condition `StringLike`. Ces qualificatifs ajoutent une fonctionnalité d'opération d'ensemble à l'opérateur de condition afin que vous puissiez tester plusieurs valeurs de demande par rapport à plusieurs valeurs de condition.

Termes connexes

- Clés de contexte à valeurs multiples
- Élément de condition

AWS politiques gérées avec cette erreur

AWS les politiques gérées vous permettent de démarrer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Les politiques AWS gérées suivantes incluent un qualificatif manquant pour les clés de condition dans leurs déclarations de politique. Lorsque vous utilisez la politique AWS gérée comme référence

pour créer votre politique gérée par le client, il est AWS recommandé d'ajouter les qualificatifs de clé de `ForAnyValue` condition `ForAllValues` ou à votre `Condition` élément.

- AWSGlueConsoleSageMakerNotebookFullAccess

Erreur : ressource manquante

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing resource: Add a Resource or NotResource element to the policy statement.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a Resource or NotResource element to the policy statement."
```

Résolution de l'erreur

Toutes les politiques, à l'exception des politiques de confiance dans les rôles, doivent inclure un `NotResource` élément `Resource` or.

Termes connexes

- Élément de ressource
- NotResource élément
- Politiques basées sur l'identité et politiques basées sur les ressources
- Présentation des politiques JSON

Erreur : instruction manquante

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing statement: Add a statement to the policy
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a statement to the policy"
```

Résolution de l'erreur

Une politique JSON doit inclure une instruction.

Termes connexes

- Éléments de politique JSON

Erreur : Null avec Si existe

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Null with if exists: The Null condition operator cannot be used with the IfExists
 suffix. Update the operator or the suffix.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The Null condition operator cannot be used with the IfExists suffix.
 Update the operator or the suffix."
```

Résolution de l'erreur

Vous pouvez ajouter `IfExists` à la fin de tous les noms d'opérateurs de condition, à l'exception de l'opérateur de condition `Null`. Utilisez un opérateur de condition `Null` pour vérifier si une clé de condition est présente lors de l'autorisation. Utilisez `...ifExists` pour spécifier que « If the policy key is present in the context of the request, process the key as specified in the policy. Si la clé n'est pas présente, évaluez l'élément de condition comme vrai. »

Termes connexes

- ... IfExists opérateurs de conditions
- Opérateur de condition null
- Élément de condition

Erreur : erreur de syntaxe SCP, caractère générique Action

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
SCP syntax error action wildcard: SCP actions can include wildcards (*) only at the end
 of a string. Update {{action}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "SCP actions can include wildcards (*) only at the end of a string.
 Update {{action}}."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des services (SCP) permettent de spécifier des valeurs dans les `NotAction` éléments `Action` or. Cependant, ces valeurs peuvent inclure des caractères génériques (*) uniquement à la fin de la chaîne. Cela signifie que vous pouvez spécifier `iam:Get*` mais pas `iam:*role`.

Pour spécifier plusieurs actions, il est AWS recommandé de les répertorier individuellement.

Termes connexes

- Action et NotAction éléments du SCP
- Évaluation du SCP
- AWS Organizations politiques de contrôle des services
- Éléments de politique JSON IAM : Action

Erreur : erreur de syntaxe SCP, autoriser Condition

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
SCP syntax error allow condition: SCPs do not support the Condition element with effect
 Allow. Update the element Condition or the effect.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "SCPs do not support the Condition element with effect Allow. Update
 the element Condition or the effect."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des services (SCP) permettent de spécifier des valeurs dans l'`Condition`élément uniquement lorsque vous l'utilisez`"Effect": "Deny"`.

Pour autoriser une seule action, vous pouvez refuser l'accès à tout, sauf à la condition que vous spécifiez à l'aide de la version `...NotEquals` d'un opérateur de condition. Cela annule la comparaison faite par l'opérateur.

Termes connexes

- Élément de condition SCP
- Évaluation du SCP
- AWS Organizations politiques de contrôle des services
- Exemple de politique : refuse l'accès AWS en fonction de la région demandée
- Éléments de politique JSON IAM : Opérateurs de condition
- Éléments de politique JSON IAM : Condition

Erreur : erreur de syntaxe SCP autorisée NotAction

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
SCP syntax error allow NotAction: SCPs do not support NotAction with effect Allow.
 Update the element NotAction or the effect.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "SCPs do not support NotAction with effect Allow. Update the element
 NotAction or the effect."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des services (SCP) ne prennent pas en charge l'utilisation de l'`NotAction`élément avec`"Effect": "Allow"`.

Vous devez réécrire la logique pour autoriser une liste d'actions ou refuser chaque action non répertoriée.

Termes connexes

- Action et NotAction éléments du SCP
- Évaluation du SCP
- AWS Organizations politiques de contrôle des services
- Éléments de politique JSON IAM : Action

Erreur : erreur de syntaxe SCP , autoriser Ressource

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
SCP syntax error allow resource: SCPs do not support Resource with effect Allow. Update
 the element Resource or the effect.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "SCPs do not support Resource with effect Allow. Update the element
 Resource or the effect."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des services (SCP) permettent de spécifier des valeurs dans l'`Resource`élément uniquement lorsque vous l'utilisez`"Effect": "Deny"`.

Vous devez réécrire la logique pour autoriser toutes les ressources ou refuser chaque ressource répertoriée.

Termes connexes

- Élément de ressource SCP
- Évaluation du SCP
- AWS Organizations politiques de contrôle des services
- Éléments de politique JSON IAM : Resource

Erreur — Erreur de syntaxe SCP NotResource

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
SCP syntax error NotResource: SCPs do not support the NotResource element. Update the
 policy to use Resource instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "SCPs do not support the NotResource element. Update the policy to
 use Resource instead."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des services (SCP) ne prennent pas en charge l'`NotResource`élément.

Vous devez réécrire la logique pour autoriser toutes les ressources ou refuser chaque ressource répertoriée.

Termes connexes

- Élément de ressource SCP
- Évaluation du SCP
- AWS Organizations politiques de contrôle des services
- Éléments de politique JSON IAM : Resource

Erreur : erreur de syntaxe SCP, principal

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
SCP syntax error principal: SCPs do not support specifying principals. Remove the
 Principal or NotPrincipal element.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "SCPs do not support specifying principals. Remove the Principal or
 NotPrincipal element."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des services (SCP) ne prennent pas en charge les `NotPrincipal` éléments `Principal` or.

Vous pouvez spécifier l'Amazon Resource Name (ARN) à l'aide de la clé de condition globale `aws:PrincipalArn` dans l'élément `Condition`.

Termes connexes

- Syntaxe d'une stratégie de contrôle de service
- Clés de condition globale pour principaux

Erreur : Sid uniques requis

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Unique Sids required: Duplicate statement IDs are not supported for this policy type.
 Update the Sid value.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Duplicate statement IDs are not supported for this policy type.
 Update the Sid value."
```

Résolution de l'erreur

Pour certains types de politique, les ID d'instruction doivent être uniques. L'élément `Sid` (ID d'instruction) vous autorise à saisir un identifiant facultatif que vous pouvez fournir pour l'instruction de politique. Vous pouvez attribuer une valeur d'ID d'instruction à chaque instruction d'un tableau d'instructions à l'aide de l'élément `SID`. Dans un service qui vous permet de spécifier un ID d'élément tel que SQS et SNS, la valeur `Sid` est un simple sous-identifiant de l'ID du document de politique. Par exemple, dans IAM, la valeur `Sid` doit être unique dans une politique JSON.

Termes connexes

- Éléments de politique JSON IAM : Sid

Erreur : action non prise en charge dans la politique

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Résolution de l'erreur

Certaines actions ne sont pas prises en charge dans l'élément `Action` dans la politique basée sur les ressources, attachée à un type de ressource différent. Par exemple, les AWS Key Management Service actions ne sont pas prises en charge dans les politiques relatives aux compartiments Amazon S3. Spécifiez une action prise en charge par le type de ressource attaché à votre politique basée sur les ressources.

Termes connexes

- Éléments de politique JSON : Action

Erreur : combinaison d'éléments non prise en charge

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Unsupported element combination: The policy elements ${element1} and ${element2} can
 not be used in the same statement. Remove one of these elements.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The policy elements ${element1} and ${element2} can not be used in
 the same statement. Remove one of these elements."
```

Résolution de l'erreur

Certaines combinaisons d'éléments de politique JSON ne peuvent pas être utilisées ensemble. Par exemple, vous ne pouvez pas utiliser `Action` et `NotAction` dans la même instruction de politique. Les autres paires qui s'excluent mutuellement sont notamment `Principal/NotPrincipal` et `Resource/NotResource`.

Termes connexes

- Référence des éléments de politique JSON IAM
- Présentation des politiques JSON

Erreur : clé de condition globale non prise en charge

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Unsupported global condition key: The condition key aws:ARN is not supported. Use
 aws:PrincipalArn or aws:SourceArn instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key aws:ARN is not supported. Use aws:PrincipalArn or
 aws:SourceArn instead."
```

Résolution de l'erreur

AWS ne prend pas en charge l'utilisation de la clé de condition globale spécifiée. En fonction de votre cas d'utilisation, vous pouvez utiliser les clés de condition globale `aws:PrincipalArn` ou `aws:SourceArn`. Par exemple, à la place de `aws:ARN`, utilisez `aws:PrincipalArn` pour comparer l'Amazon Resource Name (ARN) du principal ayant fait la demande avec l'ARN spécifié dans la politique. Vous pouvez également utiliser la clé de condition `aws:SourceArn` globale pour comparer le nom de ressource Amazon (ARN) de la ressource qui fait une service-to-service demande avec l'ARN que vous spécifiez dans la politique.

Termes connexes

- AWS clés contextuelles de condition globale

Erreur : principal non pris en charge

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Unsupported principal: The policy type ${policy_type} does not support the Principal
 element. Remove the Principal element.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The policy type ${policy_type} does not support the Principal
 element. Remove the Principal element."
```

Résolution de l'erreur

L'élément `Principal` spécifie le principal qui est autorisé ou non à accéder à une ressource. Vous ne pouvez pas utiliser l'élément `Principal` dans une politique IAM basée sur l'identité. Vous pouvez l'utiliser dans les politiques de confiance pour les rôles IAM et dans les politiques basées sur des ressources. Les politiques basées sur les ressources sont des politiques que vous intégrez directement à une ressource. Par exemple, vous pouvez intégrer des politiques dans un compartiment Amazon S3 ou dans une clé AWS KMS.

Termes connexes

- AWS Éléments de politique JSON : principal
- Accès inter-comptes aux ressources dans IAM

Erreur : ARN de ressource non pris en charge dans la politique

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The resource ARN is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Résolution de l'erreur

Certains ARN de ressources ne sont pas pris en charge dans l'élément `Resource` de la politique basée sur les ressources lorsque cette politique est associée à un autre type de ressource. Par exemple, les AWS KMS ARN ne sont pas pris en charge dans l'`Resource`élément pour les politiques

de compartiment Amazon S3. Spécifiez un ARN de ressource pris en charge par un type de ressource attaché à votre politique basée sur les ressources.

Termes connexes

- Éléments de politique JSON : Action

Erreur : sid non pris en charge

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Unsupported Sid: Update the characters in the Sid element to use one of the following
 character types: [a-z, A-Z, 0-9]
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Update the characters in the Sid element to use one of the following
 character types: [a-z, A-Z, 0-9]"
```

Résolution de l'erreur

L'élément `Sid` prend en charge les lettres majuscules, les lettres minuscules et les chiffres.

Termes connexes

- Éléments de politique JSON IAM : Sid

Erreur : caractère générique non pris en charge dans le principal

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Unsupported wildcard in principal: Wildcards (*, ?) are not supported with the
 principal key {{principal_key}}. Replace the wildcard with a valid principal value.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Wildcards (*, ?) are not supported with the principal key
 {{principal_key}}. Replace the wildcard with a valid principal value."
```

Résolution de l'erreur

La structure de l'élément `Principal` prend en charge l'utilisation d'une paire clé-valeur. La valeur du principal spécifiée dans la politique inclut un caractère générique (*). Vous ne pouvez pas inclure de caractère générique avec la clé de principal que vous avez spécifiée. Par exemple, lorsque vous spécifiez des utilisateurs dans un élément `Principal`, il n'est pas possible d'utiliser un caractère générique signifiant « tous les utilisateurs ». Vous devez nommer un ou plusieurs utilisateurs spécifiques. De même, lorsque vous spécifiez une session endossed-role, vous ne pouvez pas utiliser un caractère générique pour signifier « toutes les sessions ». Vous devez nommer une session spécifique. Vous ne pouvez pas non plus utiliser de caractère générique pour une correspondance à une partie d'un nom ou d'un ARN.

Pour résoudre ce problème, supprimez le caractère générique et fournissez un principal plus spécifique.

Termes connexes

- AWS Éléments de politique JSON : principal

Erreur : accolade manquante dans la variable

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing brace in variable: The policy variable is missing a closing curly brace. Add }
 after the variable text.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The policy variable is missing a closing curly brace. Add } after
 the variable text."
```

Résolution de l'erreur

La structure de la variable politique prend en charge l'utilisation d'un préfixe $ suivi d'une paire d'accolades (`{ }`). À l'intérieur des caractères `${ }`, incluez le nom de la valeur de la demande que vous souhaitez utiliser dans la politique.

Pour résoudre ce problème, ajoutez l'accolade manquante afin de garantir la présence de l'ensemble d'accolades d'ouverture et de fermeture.

Termes connexes

- Éléments des politiques IAM : Variables

Erreur : guillemet manquant dans la variable

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing quote in variable: The policy variable default value must begin and end with a
 single quote. Add the missing quote.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The policy variable default value must begin and end with a single
 quote. Add the missing quote."
```

Résolution de l'erreur

Lorsque vous ajoutez une variable à votre politique, vous pouvez spécifier une valeur par défaut pour la variable. Si aucune variable n'est présente, AWS utilise le texte par défaut que vous fournissez.

Pour ajouter une valeur par défaut à une variable, entourez la valeur par défaut de guillemets simples (`' '`), puis séparez le texte de la variable et la valeur par défaut par une virgule et une espace (`, `).

Par exemple, si un principal est labelisé `team=yellow`, il peut accéder au compartiment Amazon S3 `DOC-EXAMPLE-BUCKET` avec le nom `DOC-EXAMPLE-BUCKET-yellow`. Une politique avec cette ressource peut autoriser les membres de l'équipe à accéder à leurs propres ressources, mais pas à celles d'autres équipes. Pour les utilisateurs sans identifications d'équipe, vous pouvez définir une valeur par défaut de `company-wide`. Ces utilisateurs peuvent accéder uniquement au compartiment `DOC-EXAMPLE-BUCKET-company-wide`où ils peuvent afficher des informations générales, telles que des instructions pour rejoindre une équipe.

```
"Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET-${aws:PrincipalTag/team, 'company-wide'}"
```

Termes connexes

- Éléments des politiques IAM : Variables

Erreur : espace non prise en charge dans la variable

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Unsupported space in variable: A space is not supported within the policy variable
 text. Remove the space.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "A space is not supported within the policy variable text. Remove the
 space."
```

Résolution de l'erreur

La structure de la variable politique prend en charge l'utilisation d'un préfixe $ suivi d'une paire d'accolades (`{ }`). À l'intérieur des caractères `${ }`, incluez le nom de la valeur de la demande que vous souhaitez utiliser dans la politique. Bien qu'il soit possible d'inclure une espace lors de la spécification d'une variable par défaut, il n'est pas possible d'en inclure une dans le nom de la variable.

Termes connexes

- Éléments des politiques IAM : Variables

Erreur : variable vide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Empty variable: Empty policy variable. Remove the ${ } variable structure or provide a
 variable within the structure.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Empty policy variable. Remove the ${ } variable structure or provide
 a variable within the structure."
```

Résolution de l'erreur

La structure de la variable politique prend en charge l'utilisation d'un préfixe $ suivi d'une paire d'accolades ({ }). À l'intérieur des caractères ${ }, incluez le nom de la valeur de la demande que vous souhaitez utiliser dans la politique.

Termes connexes

- Éléments des politiques IAM : Variables

Erreur : variable non prise en charge dans l'élément

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Variable unsupported in element: Policy variables are supported in the Resource and
 Condition elements. Remove the policy variable {{variable}} from this element.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Policy variables are supported in the Resource and Condition
 elements. Remove the policy variable {{variable}} from this element."
```

Résolution de l'erreur

Vous pouvez utiliser des variables de politique dans l'élément `Resource` et les comparaisons de chaîne de l'élément `Condition`.

Termes connexes

- Éléments des politiques IAM : Variables

Erreur : variable non prise en charge dans la version

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Variable unsupported in version: To include variables in your policy, use the policy
 version 2012-10-17 or later.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "To include variables in your policy, use the policy version
 2012-10-17 or later."
```

Résolution de l'erreur

Pour utiliser des variables de politique, vous devez inclure l'élément `Version` et le définir sur une version qui prend en charge les variables de politique. Les variables ont été introduites dans la version `2012-10-17`. Les versions antérieures du langage de politique ne prennent pas en charge les variables de politique. Si vous ne définissez pas la `Version` sur `2012-10-17` ou ultérieure, des variables telles que `${aws:username}` sont traitées comme des chaînes littérales dans la politique.

Un élément de politique `Version` est différent d'une version de politique. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. Une version de politique est créée lorsque vous apportez des modifications à une politique gérée par le client dans IAM. La politique modifiée ne remplace pas la politique existante. À la place, IAM crée une nouvelle version de la politique gérée.

Termes connexes

- Éléments des politiques IAM : Variables
- Éléments de politique JSON IAM : Version

Erreur : adresse IP privée

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Private IP address: aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses."
```

Résolution de l'erreur

La clé de condition globale `aws:SourceIp` fonctionne uniquement pour les plages d'adresses IP publiques. Cette erreur s'affiche lorsque votre politique autorise uniquement les adresses IP privées. Dans ce cas, la condition ne correspondrait jamais.

- aws : clé de condition SourceIp globale
- Éléments de politique JSON IAM : Condition

Erreur — Privé NotIpAddress

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Private NotIpAddress: The values for condition key aws:SourceIp include only private
 IP addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The values for condition key aws:SourceIp include only private IP
 addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses."
```

Résolution de l'erreur

La clé de condition globale `aws:SourceIp` fonctionne uniquement pour les plages d'adresses IP publiques. Cette erreur s'affiche lorsque vous utilisez l'opérateur de condition `NotIpAddress` et que vous répertoriez uniquement les adresses IP privées. Dans ce cas, la condition correspondrait toujours et serait sans effet.

- aws : clé de condition SourceIp globale
- Éléments de politique JSON IAM : Condition

Erreur : la taille de la politique dépasse le quota SCP

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Policy size exceeds SCP quota: The {{policySize}} characters in the service control
 policy (SCP) exceed the {{policySizeQuota}} character maximum for SCPs. We recommend
 that you use multiple granular policies.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{policySize}} characters in the service control policy (SCP)
 exceed the {{policySizeQuota}} character maximum for SCPs. We recommend that you use
 multiple granular policies."
```

Résolution de l'erreur

AWS Organizations les politiques de contrôle des services (SCP) permettent de spécifier des valeurs dans les `NotAction` éléments `Action` or. Cependant, ces valeurs peuvent inclure des caractères génériques (*) uniquement à la fin de la chaîne. Cela signifie que vous pouvez spécifier `iam:Get*` mais pas `iam:*role`.

Pour spécifier plusieurs actions, il est AWS recommandé de les répertorier individuellement.

Termes connexes

- Quotas pour les AWS Organisations
- AWS Organizations politiques de contrôle des services

Erreur : format de principal de service non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid service principal format: The service principal does not match the expected
 format. Use the format {{expectedFormat}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The service principal does not match the expected format. Use the
 format {{expectedFormat}}."
```

Résolution de l'erreur

La valeur de la paire clé-valeur de condition doit correspondre à un format de principal de service défini.

Un principal de service est un identifiant utilisé pour accorder des autorisations à un service. Vous pouvez spécifier un principal de service dans l'élément `Principal` ou comme valeur de certaines clés de condition globale et certaines clés spécifiques au service. Le principal de service est défini par chaque service.

L'identifiant d'un principal de service inclut le nom du service, et se présente généralement sous le format suivant, en lettres minuscules :

service-name.amazonaws.com

Certaines clés spécifiques au service peuvent utiliser un format différent pour les principaux de service. Par exemple, la clé de condition `kms:ViaService` requiert le format suivant pour les principaux services, en lettres minuscules :

service-name.AWS_region.amazonaws.com

Termes connexes

- Principaux du service
- AWS clés de condition globales
- `kms:ViaService` clé de condition

Erreur : clé de balise manquante dans la condition

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing tag key in condition: The condition key {{conditionKeyName}} must include a tag
 key to control access based on tags. Use the format {{conditionKeyName}}tag-key and
 specify a key name for tag-key.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key {{conditionKeyName}} must include a tag key to
 control access based on tags. Use the format {{conditionKeyName}}tag-key and specify a
 key name for tag-key."
```

Résolution de l'erreur

Pour contrôler l'accès basé sur des balises, vous devez fournir les informations des balises dans l'élément de condition d'une politique.

Par exemple, pour contrôler l'accès aux AWS ressources, vous devez inclure la clé de `aws:ResourceTag` condition. Cette clé doit être au format `aws:ResourceTag/tag-key`. Pour spécifier la clé de balise `owner` et la valeur de balise `JaneDoe` dans une condition, utilisez le format suivant.

```
"Condition": {
    "StringEquals": {"aws:ResourceTag/owner": "JaneDoe"}
}
```

Termes connexes

- Contrôle de l'accès à l'aide de balises
- Conditions
- Clés de condition globale
- AWS clés d'état de service

Erreur : format vpc non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid vpc format: The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters."
```

Résolution de l'erreur

La clé de condition `aws:SourceVpc` doit utiliser le préfixe `vpc-` suivi de 8 ou 17 caractères alphanumériques, par exemple, `vpc-11223344556677889` ou `vpc-12345678`.

Termes connexes

- AWS clés de condition globales : aws : SourceVpc

Erreur : format vpce non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid vpce format: The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters."
```

Résolution de l'erreur

La clé de condition `aws:SourceVpce` doit utiliser le préfixe `vpce-` suivi de 8 ou 17 caractères alphanumériques, par exemple, `vpce-11223344556677889` ou `vpce-12345678`.

Termes connexes

- AWS clés de condition globales : aws : SourceVpce

Erreur : principal fédéré non pris en charge

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Federated principal not supported: The policy type does not support a federated
 identity provider in the principal element. Use a supported principal.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The policy type does not support a federated identity provider in
 the principal element. Use a supported principal."
```

Résolution de l'erreur

L'élément `Principal` utilise des principaux fédérés pour les politiques d'approbation attachées aux rôles IAM afin de fournir un accès via la fédération d'identité. Les politiques d'identité et autres

politiques basées sur les ressources ne prennent pas en charge un fournisseur d'identité fédéré dans l'élément `Principal`. Par exemple, vous ne pouvez pas utiliser de principal SAML dans une politique de compartiment Amazon S3. Modifiez l'élément `Principal` en un type principal pris en charge.

Termes connexes

- Création d'un rôle pour la fédération d'identité
- Éléments de politique JSON : principal

Erreur : action non prise en charge pour la clé de condition

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Unsupported action for condition key: The following actions: {{actions}} are not
 supported by the condition key {{key}}. The condition will not be evaluated for these
 actions. We recommend that you move these actions to a different statement without
 this condition key.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The following actions: {{actions}} are not supported by the
 condition key {{key}}. The condition will not be evaluated for these actions. We
 recommend that you move these actions to a different statement without this condition
 key."
```

Résolution de l'erreur

Assurez-vous que la clé de condition dans l'élément `Condition` de l'instruction de politique s'applique à chaque action dans l'élément `Action`. Pour vous assurer que les actions que vous spécifiez sont effectivement autorisées ou rejetées par votre politique, vous devez déplacer les actions non prises en charge vers une autre instruction sans la clé de condition.

> **Note**
>
> Si l'élément `Action` contient des actions avec des caractères génériques, IAM Access Analyzer n'évalue pas ces actions pour cette erreur.

Termes connexes

- Éléments de politique JSON : Action

Erreur : action non prise en charge dans la politique

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Résolution de l'erreur

Certaines actions ne sont pas prises en charge dans l'élément `Action` dans la politique basée sur les ressources, attachée à un type de ressource différent. Par exemple, les AWS Key Management Service actions ne sont pas prises en charge dans les politiques relatives aux compartiments Amazon S3. Spécifiez une action prise en charge par le type de ressource attaché à votre politique basée sur les ressources.

Termes connexes

- Éléments de politique JSON : Action

Erreur : ARN de ressource non pris en charge dans la politique

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The resource ARN is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Résolution de l'erreur

Certains ARN de ressources ne sont pas pris en charge dans l'élément `Resource` de la politique basée sur les ressources lorsque cette politique est associée à un autre type de ressource. Par exemple, les AWS KMS ARN ne sont pas pris en charge dans l'`Resource`élément pour les politiques de compartiment Amazon S3. Spécifiez un ARN de ressource pris en charge par un type de ressource attaché à votre politique basée sur les ressources.

Termes connexes

- Éléments de politique JSON : Action

Erreur : clé de condition non prise en charge pour le principal de service

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Unsupported condition key for service principal: The following condition keys are not
 supported when used with the service principal: {{conditionKeys}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The following condition keys are not supported when used with the
 service principal: {{conditionKeys}}."
```

Résolution de l'erreur

Vous pouvez spécifier Services AWS dans l'`Principal`élément d'une politique basée sur les ressources à l'aide d'un principal de service, qui est un identifiant du service. Vous ne pouvez pas utiliser certaines clés de condition avec certains principaux de service. Par exemple, vous ne pouvez pas utiliser la clé de condition `aws:PrincipalOrgID` avec le principal de service `cloudfront.amazonaws.com`. Vous devez supprimer les clés de condition qui ne s'appliquent pas au principal de service dans l'élément `Principal`.

Termes connexes

- Principaux du service
- Éléments de politique JSON : principal

Erreur : Erreur de syntaxe de la politique d'approbation notprincipal

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Role trust policy syntax error notprincipal: Role trust policies do not support
 NotPrincipal. Update the policy to use a Principal element instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Role trust policies do not support NotPrincipal. Update the policy
 to use a Principal element instead."
```

Résolution de l'erreur

Une politique d'approbation de rôle est une politique basée sur les ressources qui est attachée à un rôle IAM. Les politiques d'approbation définissent quelles entités de principal (comptes, utilisateurs, rôles et utilisateurs fédérés) peuvent endosser le rôle. Les politiques d'approbation de rôle ne prennent pas en charge `NotPrincipal`. Mettez à jour la politique pour utiliser un élément `Principal` à la place.

Termes connexes

- Éléments de politique JSON : principal
- Éléments de politique JSON : NotPrincipal

Erreur : politique d'approbation de rôle générique non pris en charge dans le principal

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Role trust policy unsupported wildcard in principal: "Principal:" "*" is not supported
 in the principal element of a role trust policy. Replace the wildcard with a valid
 principal value.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": ""Principal:" "*" is not supported in the principal element of a role
 trust policy. Replace the wildcard with a valid principal value."
```

Résolution de l'erreur

Une politique d'approbation de rôle est une politique basée sur les ressources qui est attachée à un rôle IAM. Les politiques d'approbation définissent quelles entités de principal (comptes, utilisateurs, rôles et utilisateurs fédérés) peuvent endosser le rôle. `"Principal:" "*"` n'est pas pris en charge dans l'élément `Principal` d'une politique d'approbation de rôle. Remplacez le caractère générique par une valeur valide du principal.

Termes connexes

- Éléments de politique JSON : principal

Erreur : Erreur de syntaxe de la politique d'approbation resource

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Role trust policy syntax error resource: Role trust policies apply to the role
 that they are attached to. You cannot specify a resource. Remove the Resource or
 NotResource element.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Role trust policies apply to the role that they are attached to. You
 cannot specify a resource. Remove the Resource or NotResource element."
```

Résolution de l'erreur

Une politique d'approbation de rôle est une politique basée sur les ressources qui est attachée à un rôle IAM. Les politiques d'approbation définissent quelles entités de principal (comptes, utilisateurs, rôles et utilisateurs fédérés) peuvent endosser le rôle. Les politiques d'approbation de rôles s'appliquent au rôle auquel elles sont associées. Vous ne pouvez pas spécifier un élément `Resource` ou `NotResource` dans une politique d'approbation de rôle. Supprimez l'élément `Resource` ou `NotResource`.

- Éléments de politique JSON : Resource

- Éléments de politique JSON : NotResource

Erreur : non-correspondance de type, plage IP

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Type mismatch IP range: The condition operator {{operator}} is used with an invalid IP
 range value. Specify the IP range in standard CIDR format.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition operator {{operator}} is used with an invalid IP range
 value. Specify the IP range in standard CIDR format."
```

Résolution de l'erreur

Mettez à jour le texte pour utiliser le type de données de l'opérateur de condition Adresse IP, au format CIDR.

Termes connexes

- Opérateurs de condition d'adresse IP
- Éléments de politique JSON IAM : Opérateurs de condition

Erreur : action manquante pour la clé de condition

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing action for condition key: The {{actionName}} action must be in the action block
 to allow setting values for the condition key {{keyName}}. Add {{actionName}} to the
 action block.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{actionName}} action must be in the action block to allow
 setting values for the condition key {{keyName}}. Add {{actionName}} to the action
 block."
```

Résolution de l'erreur

La clé de condition dans l'élément `Condition` de la déclaration de politique n'est pas évalué à moins que l'action spécifiée ne figure dans l'élément `Action`. Pour vous assurer que les clés de condition que vous spécifiez sont effectivement autorisées ou rejetées par votre politique, ajoutez l'action à l'élément `Action`.

Termes connexes

- Éléments de politique JSON : Action

Erreur : Syntaxe du principal fédéré non valide dans la politique d'approbation de rôle

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid federated principal syntax in role trust policy: The principal value specifies
 a federated principal that does not match the expected format. Update the federated
 principal to a domain name or a SAML metadata ARN.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The principal value specifies a federated principal that does not
 match the expected format. Update the federated principal to a domain name or a SAML
 metadata ARN."
```

Résolution de l'erreur

La valeur du principal indique un principal fédéré qui ne correspond pas au format attendu. Mettez à jour le format du principal fédéré en utilisant un nom de domaine valide ou un ARN de métadonnées SAML.

Termes connexes

- Utilisateurs fédérés et rôles

Erreur : action non correspondante pour le principal

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Mismatched action for principal: The {{actionName}} action is invalid with the
 following principal(s): {{principalNames}}. Use a SAML provider principal with
 the sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{actionName}} action is invalid with the following
 principal(s): {{principalNames}}. Use a SAML provider principal with the
 sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options."
```

Résolution de l'erreur

L'action spécifiée dans l'élément `Action` de la déclaration de politique n'est pas valide avec le principal spécifié dans l'élément `Principal`. Par exemple, vous ne pouvez pas utiliser de principal de fournisseur SAML avec l'action `sts:AssumeRoleWithWebIdentity`. Vous devez utiliser un principal de fournisseur SAML avec l'action `sts:AssumeRoleWithSAML` ou un principal de fournisseur OIDC avec l'action `sts:AssumeRoleWithWebIdentity`.

Termes connexes

- AssumeRoleWithSAML
- AssumeRoleWithWebIdentity

Erreur : action manquante pour la politique d'approbation roles anywhere

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing action for roles anywhere trust policy: The rolesanywhere.amazonaws.com service
 principal requires the sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession
 permissions to assume a role. Add the missing permissions to the policy.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The rolesanywhere.amazonaws.com service principal requires the
 sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession permissions to assume a
 role. Add the missing permissions to the policy."
```

Résolution de l'erreur

Pour que Rôles Anywhere IAM soit en mesure d'assumer un rôle et de fournir des informations d'identification AWS temporaires, le rôle doit approuver le principal de service Rôles Anywhere IAM. Le principal de service Rôles Anywhere IAM nécessite les autorisations `sts:AssumeRole`, `sts:SetSourceIdentity`, et `sts:TagSession` pour assumer un rôle. Si l'une des autorisations est manquante, vous devez l'ajouter à votre politique.

Termes connexes

- Modèle de confiance dans AWS Identity and Access Management Roles Anywhere

Avertissement général — Créez un reflex avec NotResource

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Create SLR with NotResource: Using the iam:CreateServiceLinkedRole action with
 NotResource can allow creation of unintended service-linked roles for multiple
 resources. We recommend that you specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with NotResource can
 allow creation of unintended service-linked roles for multiple resources. We recommend
 that you specify resource ARNs instead."
```

Résolution de l'avertissement général

L'action `iam:CreateServiceLinkedRole` autorise la création d'un rôle IAM qui permet à un AWS service d'effectuer des actions en votre nom. L'utilisation de `iam:CreateServiceLinkedRole` dans une politique avec l'élément `NotResource` peut autoriser la création de rôles liés au service inattendus pour plusieurs ressources. AWS vous recommande plutôt de spécifier les ARN autorisés dans l'élément `Resource`.

- CreateServiceLinkedRole opération

- Éléments de politique JSON IAM : NotResource
- Éléments de politique JSON IAM : Resource

Avertissement général — Créez un reflex avec une étoile en action et NotResource

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Create SLR with star in action and NotResource: Using an action with a wildcard(*) and
 NotResource can allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using an action with a wildcard(*) and NotResource can
 allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead."
```

Résolution de l'avertissement général

L'action `iam:CreateServiceLinkedRole` autorise la création d'un rôle IAM qui permet à un AWS service d'effectuer des actions en votre nom. Les politiques comportant un caractère générique (*) dans le champ `Action`, et comprenant l'élément `NotResource` peuvent autoriser la création de rôles liés au service inattendus pour plusieurs ressources. AWS vous recommande plutôt de spécifier les ARN autorisés dans l'élément `Resource`.

- CreateServiceLinkedRole opération
- Éléments de politique JSON IAM : NotResource
- Éléments de politique JSON IAM : Resource

Avertissement général — Créez un reflex avec et NotAction NotResource

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Create SLR with NotAction and NotResource: Using NotAction with NotResource
 can allow creation of unintended service-linked roles because it allows
```

```
iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using NotAction with NotResource can allow creation of unintended
 service-linked roles because it allows iam:CreateServiceLinkedRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead."
```

Résolution de l'avertissement général

L'action `iam:CreateServiceLinkedRole` autorise la création d'un rôle IAM qui permet à un AWS service d'effectuer des actions en votre nom. L'utilisation de l'élément `NotAction` avec l'élément `NotResource` peut autoriser la création de rôles liés au service inattendus pour plusieurs ressources. AWS vous recommande plutôt de réécrire la politique afin d'autoriser `iam:CreateServiceLinkedRole` sur une liste limitée d'ARN dans l'élément `Resource`. Vous pouvez également ajouter `iam:CreateServiceLinkedRole` à l'élément `NotAction`.

- CreateServiceLinkedRole opération
- Éléments de politique JSON IAM : NotAction
- Éléments de politique JSON IAM : Action
- Éléments de politique JSON IAM : NotResource
- Éléments de politique JSON IAM : Resource

Avertissement général : créer un SLR avec star in resource

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Create SLR with star in resource: Using the iam:CreateServiceLinkedRole action with
 wildcards (*) in the resource can allow creation of unintended service-linked roles.
 We recommend that you specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with wildcards (*) in
 the resource can allow creation of unintended service-linked roles. We recommend that
 you specify resource ARNs instead."
```

Résolution de l'avertissement général

L'action `iam:CreateServiceLinkedRole` autorise la création d'un rôle IAM qui permet à un AWS service d'effectuer des actions en votre nom. L'utilisation de `iam:CreateServiceLinkedRole` dans une politique avec le caractère générique (*) dans l'élément `Resource` peut autoriser la création de rôles liés au service inattendus pour plusieurs ressources. AWS vous recommande plutôt de spécifier les ARN autorisés dans l'élément `Resource`.

- CreateServiceLinkedRole opération
- Éléments de politique JSON IAM : Resource

AWS politiques gérées avec cet avertissement général

AWS les politiques gérées vous permettent de démarrer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Certains de ces cas d'utilisation sont destinés aux utilisateurs « avec pouvoir » dans votre compte. Les politiques AWS gérées suivantes fournissent un accès aux utilisateurs expérimentés et accordent des autorisations pour créer des rôles liés à un service pour n'importe quel AWS service. AWS recommande d'associer les politiques AWS gérées suivantes uniquement aux identités IAM que vous considérez comme des utilisateurs expérimentés.

- PowerUserAccess
- AlexaForBusinessFullAccess
- AWSOrganizationsServiceTrustPolicy— Cette politique AWS gérée fournit des autorisations d'utilisation par le rôle AWS Organizations lié au service. Ce rôle permet aux Organisations de créer des rôles supplémentaires liés aux services pour d'autres services de votre AWS organisation.

Avertissement général : créer un SLR avec star in action and resource

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Create SLR with star in action and resource: Using wildcards (*) in the action and
 the resource can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using wildcards (*) in the action and the resource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead."
```

Résolution de l'avertissement général

L'action `iam:CreateServiceLinkedRole` autorise la création d'un rôle IAM qui permet à un AWS service d'effectuer des actions en votre nom. Les politiques comportant un caractère générique (*) dans les éléments `Action` et `Resource` peuvent autoriser la création de rôles liés au service inattendus pour plusieurs ressources. Cela permet de créer un rôle lié à un service lorsque vous spécifiez `"Action": "*""Action": "iam:*"`, ou. `"Action": "iam:Create*"` AWS recommande de spécifier plutôt les ARN autorisés dans l'`Resource`élément.

- CreateServiceLinkedRole opération
- Éléments de politique JSON IAM : Action
- Éléments de politique JSON IAM : Resource

AWS politiques gérées avec cet avertissement général

AWS les politiques gérées vous permettent de démarrer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Certains de ces cas d'utilisation sont destinés aux administrateurs de votre compte. Les politiques AWS gérées suivantes fournissent un accès administrateur et accordent des autorisations pour créer des rôles liés à un service pour n'importe quel AWS service. AWS recommande d'associer les politiques AWS gérées suivantes uniquement aux identités IAM que vous considérez comme des administrateurs.

- AdministratorAccess

- IAM FullAccess

Avertissement général — Créez un reflex avec une étoile dans la ressource et NotAction

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Create SLR with star in resource and NotAction: Using a resource with wildcards (*)
 and NotAction can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can allow creation
 of unintended service-linked roles because it allows iam:CreateServiceLinkedRole
 permissions on all resources. We recommend that you specify resource ARNs instead."
```

Résolution de l'avertissement général

L'action `iam:CreateServiceLinkedRole` autorise la création d'un rôle IAM qui permet à un AWS service d'effectuer des actions en votre nom. L'utilisation de l'élément `NotAction` dans une politique avec le caractère générique (*) dans l'élément `Resource` peut autoriser la création de rôles liés au service inattendus pour plusieurs ressources. AWS vous recommande plutôt de spécifier les ARN autorisés dans l'élément `Resource`. Vous pouvez également ajouter `iam:CreateServiceLinkedRole` à l'élément `NotAction`.

- CreateServiceLinkedRole opération
- Éléments de politique JSON IAM : NotAction
- Éléments de politique JSON IAM : Action
- Éléments de politique JSON IAM : Resource

Avertissement général : clé de condition globale obsolète

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Deprecated global condition key: We recommend that you update aws:ARN to use the newer
 condition key aws:PrincipalArn.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "We recommend that you update aws:ARN to use the newer condition key
 aws:PrincipalArn."
```

Résolution de l'avertissement général

La politique inclut une clé de condition globale obsolète. Mettez à jour la clé de condition dans la paire clé-valeur de condition pour utiliser une clé de condition globale prise en charge.

- Clés de condition globale

Avertissement général : valeur de date non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid date value: The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format."
```

Résolution de l'avertissement général

L'heure d'une époque Unix décrit un point dans le temps qui s'est écoulé depuis le 1er janvier 1970, moins les secondes intercalaires. Il se peut que le temps d'Epoch ne se résorbe pas à l'heure précise à laquelle vous vous attendiez. AWS recommande d'utiliser le standard W3C pour les formats de date et d'heure. Par exemple, vous pouvez spécifier une date complète, telle que YYYY-MM-DD (1997-07-16), ou vous pouvez également ajouter l'heure à la seconde, telle que YYYY-MM-DDThh:mm:ssTZD (1997-07-16T19:20:30+01:00).

- Formats de date et d'heure selon la norme W3C
- Éléments de politique JSON IAM : Version
- aws : clé de condition CurrentTime globale

Avertissement général : référence de rôle non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid role reference: The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead."
```

Résolution de l'avertissement général

AWS vous recommande de spécifier l'Amazon Resource Name (ARN) pour un rôle IAM au lieu de son ID principal. Lorsque IAM enregistre la politique, il transforme l'ARN en identifiant principal pour le rôle existant. AWS inclut une mesure de sécurité. Si quelqu'un supprime le rôle et le recrée, il aura un nouvel ID, et la politique ne correspondra pas à l'ID du nouveau rôle.

- Spécification d'un principal : rôles IAM
- ARN IAM
- ID uniques IAM

Avertissement général : référence utilisateur non valide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Invalid user reference: The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead."
```

Résolution de l'avertissement général

AWS vous recommande de spécifier l'Amazon Resource Name (ARN) pour un utilisateur IAM au lieu de son ID principal. Lorsque IAM enregistre la politique, il transforme l'ARN en identifiant principal de l'utilisateur existant. AWS inclut une mesure de sécurité. Si quelqu'un supprime l'utilisateur et le recrée, il aura un nouvel ID, et la politique ne correspondra pas à l'ID du nouvel utilisateur.

- Spécification d'un principal : utilisateurs IAM
- ARN IAM
- ID uniques IAM

Avertissement général : version manquante

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing version: We recommend that you specify the Version element to help you with
 debugging permission issues.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "We recommend that you specify the Version element to help you with
 debugging permission issues."
```

Résolution de l'avertissement général

AWS vous recommande d'inclure le `Version` paramètre facultatif dans votre politique. Si vous n'incluez pas d'élément Version, la valeur par défaut est `2012-10-17`, mais de nouvelles fonctions, telles que des variables de politique, ne fonctionneront pas avec votre politique. Par exemple, les variables telles que `${aws:username}` ne sont pas reconnues et sont traitées comme des chaînes littérales dans la politique.

- Éléments de politique JSON IAM : Version

Avertissement général : sid uniques recommandés

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Unique Sids recommended: We recommend that you use statement IDs that are unique to
 your policy. Update the Sid value.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "We recommend that you use statement IDs that are unique to your
 policy. Update the Sid value."
```

Résolution de l'avertissement général

AWS vous recommande d'utiliser des identifiants de relevé uniques. L'élément `Sid` (ID d'instruction) vous autorise à saisir un identifiant facultatif que vous pouvez fournir pour l'instruction de politique. Vous pouvez attribuer une valeur d'ID d'instruction à chaque instruction d'un tableau d'instructions à l'aide de l'élément `SID`.

Termes connexes

- Éléments de politique JSON IAM : Sid

Avertissement général : caractère générique sans opérateur similaire

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Wildcard without like operator: Your condition value includes a * or ? character. If
 you meant to use a wildcard (*, ?), update the condition operator to include Like.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Your condition value includes a * or ? character. If you meant to
 use a wildcard (*, ?), update the condition operator to include Like."
```

Résolution de l'avertissement général

La structure de l'élément `Condition` a besoin que vous utilisiez un opérateur de condition et une paire clé-valeur. Lorsque vous spécifiez une valeur de condition qui utilise un caractère générique (*, ?), vous devez utiliser la version `Like` de l'opérateur de condition. Par exemple, au lieu de l'opérateur de condition de chaîne `StringEquals`, utilisez `StringLike`.

```
"Condition": {"StringLike": {"aws:PrincipalTag/job-category": "admin-*"}}
```

- Éléments de politique JSON IAM : Opérateurs de condition
- Éléments de politique JSON IAM : Condition

AWS politiques gérées avec cet avertissement général

AWS les politiques gérées vous permettent de démarrer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Les politiques AWS gérées suivantes incluent des caractères génériques dans leur valeur de condition sans opérateur de condition qui inclut la correspondance `Like` de modèles. Lorsque vous utilisez la politique AWS gérée comme référence pour créer votre politique gérée par le client, il est AWS recommandé d'utiliser un opérateur de condition qui prend en charge la correspondance de modèles avec des caractères génériques (*, ?) , tels que`StringLike`.

- AWSGlueConsoleSageMakerNotebookFullAccess

Avertissement général : la taille de la politique dépasse le quota de politique

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Policy size exceeds identity policy quota: The {{policySize}} characters in the
 identity policy, excluding whitespace, exceed the {{policySizeQuota}} character
 maximum for inline and managed policies. We recommend that you use multiple granular
 policies.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{policySize}} characters in the identity policy, excluding
 whitespace, exceed the {{policySizeQuota}} character maximum for inline and managed
 policies. We recommend that you use multiple granular policies."
```

Résolution de l'avertissement général

Vous pouvez attacher jusqu'à 10 politiques gérées à une identité IAM (utilisateur, groupe d'utilisateurs ou rôle). La taille de chaque politique gérée ne peut toutefois pas dépasser le quota par défaut de 6 144 caractères. IAM ne compte pas les espaces lors du calcul de la taille d'une politique par rapport à ce quota. Les quotas, également appelés limites dans AWS, sont les valeurs maximales pour les ressources, les actions et les éléments de votre AWS compte.

Vous pouvez également ajouter autant de politiques en ligne que vous le voulez à une identité IAM. Cependant, la somme de toutes les politiques en ligne par identité ne peut pas dépasser le quota spécifié.

Si votre politique dépasse le quota, vous pouvez organiser votre politique en plusieurs instructions et les regrouper en plusieurs politiques.

Termes connexes

- IAM et quotas de AWS STS caractères
- Plusieurs instructions et politiques
- Politiques gérées par le client IAM
- Présentation des politiques JSON
- Syntaxe de politique IAM JSON

AWS politiques gérées avec cet avertissement général

AWS les politiques gérées vous permettent de démarrer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Les politiques AWS gérées suivantes accordent des autorisations pour des actions sur de nombreux AWS services et dépassent la taille maximale des politiques. Lorsque vous utilisez la politique gérée par AWS comme référence pour créer votre politique gérée, vous devez diviser la politique en plusieurs politiques.

- ReadOnlyAccess
- AWSSupportServiceRolePolicy

Avertissement général : la taille de la politique dépasse le quota de la politique de ressources

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Policy size exceeds resource policy quota: The {{policySize}} characters in the
 resource policy exceed the {{policySizeQuota}} character maximum for resource
 policies. We recommend that you use multiple granular policies.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{policySize}} characters in the resource policy exceed the
 {{policySizeQuota}} character maximum for resource policies. We recommend that you use
 multiple granular policies."
```

Résolution de l'avertissement général

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource, telle qu'un compartiment Amazon S3. Ces politiques accordent au principal spécifié l'autorisation d'effectuer des actions spécifiques sur cette ressource et définissent sous quelles conditions cela s'applique. La taille des politiques basées sur les ressources ne peut pas dépasser le quota défini pour cette ressource. Les quotas, également appelés limites dans AWS, sont les valeurs maximales pour les ressources, les actions et les éléments de votre AWS compte.

Si votre politique dépasse le quota, vous pouvez organiser votre politique en plusieurs instructions et les regrouper en plusieurs politiques.

Termes connexes

- Politiques basées sur les ressources
- Politiques de compartiment Amazon S3
- Plusieurs instructions et politiques
- Présentation des politiques JSON
- Syntaxe de politique IAM JSON

Avertissement général : non-correspondance de type

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Type mismatch: Use the operator type {{allowed}} instead of operator {{operator}} for
 the condition key {{key}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Use the operator type {{allowed}} instead of operator {{operator}}
 for the condition key {{key}}."
```

Résolution de l'avertissement général

Mettez à jour le texte pour utiliser le type de données de l'opérateur de condition pris en charge.

Par exemple, la clé de condition globale `aws:MultiFactorAuthPresent` a besoin d'un opérateur de condition avec le type de données `Boolean`. Si vous fournissez une date ou un entier, le type de données ne correspondra pas.

Termes connexes

- Clés de condition globale
- Éléments de politique JSON IAM : Opérateurs de condition

Avertissement général : non-correspondance de type, opérateur booléen

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Type mismatch Boolean: Add a valid Boolean value (true or false) for the condition
 operator {{operator}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a valid Boolean value (true or false) for the condition operator
 {{operator}}."
```

Résolution de l'avertissement général

Mettez à jour le texte pour utiliser un type de données d'opérateur de condition booléen, tel que `true` ou `false`.

Par exemple, la clé de condition globale `aws:MultiFactorAuthPresent` a besoin d'un opérateur de condition avec le type de données `Boolean`. Si vous fournissez une date ou un entier, le type de données ne correspondra pas.

Termes connexes

- Opérateurs de condition booléens
- Éléments de politique JSON IAM : Opérateurs de condition

Avertissement général : non-correspondance de type, date

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Type mismatch date: The date condition operator is used with an invalid value. Specify
 a valid date using YYYY-MM-DD or other ISO 8601 date/time format.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The date condition operator is used with an invalid value. Specify a
 valid date using YYYY-MM-DD or other ISO 8601 date/time format."
```

Résolution de l'avertissement général

Mettez à jour le texte pour utiliser le type de données de l'opérateur de condition Date, dans un format d'heure/date `YYYY-MM-DD` ou un autre format d'heure/date ISO 8601.

Termes connexes

- Opérateurs de condition de date
- Éléments de politique JSON IAM : Opérateurs de condition

Avertissement général : numéro de non-correspondance de type

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Type mismatch number: Add a valid numeric value for the condition operator
 {{operator}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a valid numeric value for the condition operator {{operator}}."
```

Résolution de l'avertissement général

Mettez à jour le texte pour utiliser le type de données de l'opérateur de condition numérique.

Termes connexes

- Opérateurs de condition numériques
- Éléments de politique JSON IAM : Opérateurs de condition

Avertissement général : non-correspondance de type, chaîne

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Type mismatch string: Add a valid base64-encoded string value for the condition
 operator {{operator}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a valid base64-encoded string value for the condition operator
 {{operator}}."
```

Résolution de l'avertissement général

Mettez à jour le texte pour utiliser le type de données de l'opérateur de condition Chaîne.

Termes connexes

- Opérateurs de condition de chaîne
- Éléments de politique JSON IAM : Opérateurs de condition

Avertissement général : github repo et branch spécifiques recommandés

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Specific github repo and branch recommended: Using a wildcard (*) in
 token.actions.githubusercontent.com:sub can allow requests from more sources than
 you intended. Specify the value of token.actions.githubusercontent.com:sub with the
 repository and branch name.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using a wildcard (*) in token.actions.githubusercontent.com:sub
 can allow requests from more sources than you intended. Specify the value of
 token.actions.githubusercontent.com:sub with the repository and branch name."
```

Résolution de l'avertissement général

Si vous l'utilisez GitHub en tant qu'IdP OIDC, la meilleure pratique consiste à limiter les entités qui peuvent assumer le rôle associé à l'IdP IAM. Lorsque vous incluez une `Condition` déclaration dans une politique de confiance des rôles, vous pouvez limiter le rôle à une GitHub organisation, un référentiel ou une branche spécifique. Vous pouvez utiliser la clé de condition `token.actions.githubusercontent.com:sub` pour limiter l'accès. Nous vous recommandons de limiter la condition à un ensemble spécifique de référentiels ou de branches. Si vous utilisez un caractère générique (`*`) dans`token.actions.githubusercontent.com:sub`, les GitHub actions provenant d'organisations ou de référentiels indépendants de votre volonté peuvent assumer les rôles associés à l'IdP GitHub IAM dans votre compte. AWS

Termes connexes

- Configuration d'un rôle pour le fournisseur d'identité GitHub OIDC

Avertissement général : la taille de la politique dépasse le quota de la politique de rôle

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Policy size exceeds role trust policy quota: The characters in the role trust policy,
 excluding whitespace, exceed the character maximum. We recommend that you request a
 role trust policy length quota increase using Service Quotas and AWS Support Center.
 If the quotas have already been increased, then you can ignore this warning.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The characters in the role trust policy, excluding whitespace,
 exceed the character maximum. We recommend that you request a role trust policy length
 quota increase using Service Quotas and AWS Support Center. If the quotas have already
 been increased, then you can ignore this warning."
```

Résolution de l'avertissement général

IAM et AWS STS disposez de quotas qui limitent la taille des politiques de confiance dans les rôles. Les caractères de la politique d'approbation de rôle, à l'exception des espaces, dépassent le nombre maximum de caractères. Nous vous recommandons de demander une augmentation du quota de durée de la politique d'approbation de rôle en utilisant Service Quotas et AWS Support Center Console.

Termes connexes

- IAM et AWS STS quotas, exigences relatives aux noms et limites de caractères

Avertissement de sécurité — Autoriser avec NotPrincipal

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Allow with NotPrincipal: Using Allow with NotPrincipal can be overly permissive. We
 recommend that you use Principal instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using Allow with NotPrincipal can be overly permissive. We recommend
 that you use Principal instead."
```

Résolution de l'avertissement de sécurité

L'utilisation de `"Effect": "Allow"` avec `NotPrincipal` peut être trop permissive. Par exemple, cela peut accorder des autorisations à des mandants anonymes. AWS vous recommande de spécifier les principaux qui ont besoin d'un accès à l'aide de l'`Principal`élément. En variante, vous pouvez autoriser un accès étendu, puis ajouter une autre instruction qui utilise l'élément `NotPrincipal` avec `“Effect”: “Deny”`.

- AWS Éléments de politique JSON : principal
- AWS Éléments de politique JSON : NotPrincipal

Avertissement de sécurité — ForAllValues avec clé à valeur unique

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
ForAllValues with single valued key: Using ForAllValues qualifier with the single-
valued condition key {{key}} can be overly permissive. We recommend that you remove
 ForAllValues:.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using ForAllValues qualifier with the single-valued condition key
 {{key}} can be overly permissive. We recommend that you remove ForAllValues:."
```

Résolution de l'avertissement de sécurité

AWS recommande de n'utiliser `ForAllValues` que les conditions à valeurs multiples. L'opération d'ensemble `ForAllValues` teste si la valeur de chaque membre de la requête est un sous-ensemble de la clé de condition. La condition renvoie la valeur true si chaque valeur de clé de la demande correspond à au moins une valeur de la politique. Elle renvoie également la valeur Vrai si la demande ne comprend pas de clés ou si les valeurs de clé aboutissent à un ensemble de données nul, tel qu'une chaîne vide.

Pour savoir si une condition prend en charge une valeur unique ou plusieurs valeurs, passez en revue la page Actions, ressources et clés de condition pour le service. Les clés de condition avec le préfixe de type de données `ArrayOf` sont des clés de condition à valeurs multiples. Par exemple, Amazon SES prend en charge les clés avec des valeurs uniques (`String`) et le type de données à valeurs multiples `ArrayOfString`.

- Clés de contexte à valeurs multiples

Avertissement de sécurité — Transmettez le rôle avec NotResource

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Pass role with NotResource: Using the iam:PassRole action with NotResource can be
 overly permissive because it can allow iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the iam:PassRole action with NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

Résolution de l'avertissement de sécurité

Pour configurer de nombreux AWS services, vous devez leur transmettre un rôle IAM. Pour permettre cela, vous devez accorder l'autorisation `iam:PassRole` à une identité (utilisateur, groupe d'utilisateurs ou rôle). L'utilisation de `NotResource` cet élément `iam:PassRole` dans une politique peut permettre à vos mandants d'accéder à un plus grand nombre de services ou de fonctionnalités que vous ne le souhaitiez. AWS recommande de spécifier plutôt les ARN autorisés dans l'`Resource`élément. En outre, vous pouvez réduire les autorisations à un seul service en utilisant la clé de condition `iam:PassedToService`.

- Transférer un rôle à un service
- iam : PassedToService
- Éléments de politique JSON IAM : NotResource
- Éléments de politique JSON IAM : Resource

Avertissement de sécurité — Passez le rôle avec l'étoile en action et NotResource

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Pass role with star in action and NotResource: Using an action with a wildcard (*) and
 NotResource can be overly permissive because it can allow iam:PassRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using an action with a wildcard (*) and NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

Résolution de l'avertissement de sécurité

Pour configurer de nombreux AWS services, vous devez leur transmettre un rôle IAM. Pour permettre cela, vous devez accorder l'autorisation `iam:PassRole` à une identité (utilisateur, groupe

d'utilisateurs ou rôle). Les politiques comportant un caractère générique (*) dans le `Action` et qui incluent l'`NotResource`élément peuvent permettre à vos mandants d'accéder à un plus grand nombre de services ou de fonctionnalités que vous ne le souhaitiez. AWS recommande de spécifier plutôt les ARN autorisés dans l'`Resource`élément. En outre, vous pouvez réduire les autorisations à un seul service en utilisant la clé de condition `iam:PassedToService`.

- Transférer un rôle à un service
- iam : PassedToService
- Éléments de politique JSON IAM : NotResource
- Éléments de politique JSON IAM : Resource

Avertissement de sécurité — Transmettez le rôle avec NotAction et NotResource

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Pass role with NotAction and NotResource: Using NotAction with NotResource can
 be overly permissive because it can allow iam:PassRole permissions on multiple
 resources.. We recommend that you specify resource ARNs instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using NotAction with NotResource can be overly permissive because
 it can allow iam:PassRole permissions on multiple resources.. We recommend that you
 specify resource ARNs instead."
```

Résolution de l'avertissement de sécurité

Pour configurer de nombreux AWS services, vous devez leur transmettre un rôle IAM. Pour permettre cela, vous devez accorder l'autorisation `iam:PassRole` à une identité (utilisateur, groupe d'utilisateurs ou rôle). L'utilisation de l'`NotAction`élément et la `NotResource` liste de certaines ressources qu'il contient peuvent permettre à vos principaux utilisateurs d'accéder à plus de services ou de fonctionnalités que prévu. AWS recommande de spécifier plutôt les ARN autorisés dans l'`Resource`élément. En outre, vous pouvez réduire les autorisations à un seul service en utilisant la clé de condition `iam:PassedToService`.

- Transférer un rôle à un service
- iam : PassedToService

- Éléments de politique JSON IAM : NotAction
- Éléments de politique JSON IAM : Action
- Éléments de politique JSON IAM : NotResource
- Éléments de politique JSON IAM : Resource

Avertissement de sécurité : transférer le rôle avec étoile dans la ressource

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Pass role with star in resource: Using the iam:PassRole action with wildcards (*)
 in the resource can be overly permissive because it allows iam:PassRole permissions
 on multiple resources. We recommend that you specify resource ARNs or add the
 iam:PassedToService condition key to your statement.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the iam:PassRole action with wildcards (*) in the resource can
 be overly permissive because it allows iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs or add the iam:PassedToService condition
 key to your statement."
```

Résolution de l'avertissement de sécurité

Pour configurer de nombreux AWS services, vous devez leur transmettre un rôle IAM. Pour permettre cela, vous devez accorder l'autorisation `iam:PassRole` à une identité (utilisateur, groupe d'utilisateurs ou rôle). Les politiques qui autorisent `iam:PassRole` et incluent un caractère générique (*) dans l'`Resource`élément peuvent permettre à vos principaux utilisateurs d'accéder à un plus grand nombre de services ou de fonctionnalités que vous ne le souhaitiez. AWS recommande de spécifier plutôt les ARN autorisés dans l'`Resource`élément. En outre, vous pouvez réduire les autorisations à un seul service en utilisant la clé de condition `iam:PassedToService`.

Certains AWS services incluent leur espace de noms de service dans le nom de leur rôle. Cette vérification de politique tient compte de ces conventions lors de l'analyse de la politique afin de générer des conclusions. Par exemple, l'ARN de ressource suivant peut ne pas générer de résultat :

```
arn:aws:iam::*:role/Service*
```

- Transférer un rôle à un service
- iam : PassedToService
- Éléments de politique JSON IAM : Resource

AWS politiques gérées avec cet avertissement de sécurité

AWS les politiques gérées vous permettent de démarrer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

L'un de ces cas d'utilisation concerne les administrateurs de votre compte. Les politiques AWS gérées suivantes fournissent un accès administrateur et accordent des autorisations pour transmettre un rôle IAM à n'importe quel service. AWS recommande d'associer les politiques AWS gérées suivantes uniquement aux identités IAM que vous considérez comme des administrateurs.

- AdministratorAccess-Amplifier

Les politiques AWS gérées suivantes incluent des autorisations permettant `iam:PassRole` d'utiliser un caractère générique (*) dans la ressource et sont vouées à l'obsolescence. Pour chacune de ces politiques, nous avons mis à jour les directives relatives aux autorisations, notamment en recommandant une nouvelle politique AWS gérée qui prend en charge le cas d'utilisation. Pour afficher des alternatives à ces politiques, veuillez consulter les guides de chaque service.

- AWSElasticBeanstalkFullAccess
- AWSElasticBeanstalkService
- AWSLambdaFullAccess
- AWSLambdaReadOnlyAccess
- AWSOpsWorksFullAccess
- AWSOpsWorksRole
- AWSDataPipelineRole
- AmazonDynamoDB FullAccesswithDataPipeline
- AmazonElasticMapReduceFullAccess
- AmazonDynamoDB FullAccesswithDataPipeline
- Amazon EC2 ContainerServiceFullAccess

Les politiques AWS gérées suivantes fournissent des autorisations uniquement pour les rôles liés aux services, ce qui permet aux AWS services d'effectuer des actions en votre nom. Vous ne pouvez pas attacher ces politiques à vos identités IAM.

- AWSServiceRoleForAmazonEKSNodegroup

Avertissement de sécurité : transférer le rôle avec étoile dans l'action et la ressource

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Pass role with star in action and resource: Using wildcards (*) in the action and the
 resource can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using wildcards (*) in the action and the resource can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
 that you specify resource ARNs or add the iam:PassedToService condition key to your
 statement."
```

Résolution de l'avertissement de sécurité

Pour configurer de nombreux AWS services, vous devez leur transmettre un rôle IAM. Pour permettre cela, vous devez accorder l'autorisation `iam:PassRole` à une identité (utilisateur, groupe d'utilisateurs ou rôle). Les politiques comportant un caractère générique (*) dans les `Resource` éléments `Action` et peuvent permettre à vos mandants d'accéder à un plus grand nombre de services ou de fonctionnalités que vous ne le souhaitiez. AWS recommande de spécifier plutôt les ARN autorisés dans l'`Resource`élément. En outre, vous pouvez réduire les autorisations à un seul service en utilisant la clé de condition `iam:PassedToService`.

- Transférer un rôle à un service
- iam : PassedToService
- Éléments de politique JSON IAM : Action
- Éléments de politique JSON IAM : Resource

AWS politiques gérées avec cet avertissement de sécurité

AWS les politiques gérées vous permettent de démarrer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Certains de ces cas d'utilisation sont destinés aux administrateurs de votre compte. Les politiques AWS gérées suivantes fournissent un accès administrateur et accordent des autorisations pour transmettre un rôle IAM à n'importe quel AWS service. AWS recommande d'associer les politiques AWS gérées suivantes uniquement aux identités IAM que vous considérez comme des administrateurs.

- AdministratorAccess
- IAM FullAccess

Avertissement de sécurité — Passez le rôle avec une étoile dans la ressource et NotAction

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Pass role with star in resource and NotAction: Using a resource with wildcards (*) and
 NotAction can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
 that you specify resource ARNs or add the iam:PassedToService condition key to your
 statement."
```

Résolution de l'avertissement de sécurité

Pour configurer de nombreux AWS services, vous devez leur transmettre un rôle IAM. Pour permettre cela, vous devez accorder l'autorisation `iam:PassRole` à une identité (utilisateur, groupe d'utilisateurs ou rôle). L'utilisation de l'`NotAction`élément dans une politique avec un caractère générique (*) dans l'`Resource`élément peut permettre à vos principaux utilisateurs d'accéder à plus de services ou de fonctionnalités que vous ne le souhaitiez. AWS recommande de spécifier plutôt les

ARN autorisés dans l'`Resource`élément. En outre, vous pouvez réduire les autorisations à un seul service en utilisant la clé de condition `iam:PassedToService`.

- Transférer un rôle à un service
- iam : PassedToService
- Éléments de politique JSON IAM : NotAction
- Éléments de politique JSON IAM : Action
- Éléments de politique JSON IAM : Resource

Avertissement de sécurité : clés de condition appariées manquantes

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing paired condition keys: Using the condition key {{conditionKeyName}}
 can be overly permissive without also using the following condition keys:
 {{recommendedKeys}}. Condition keys like this one are more secure when paired with
 a related key. We recommend that you add the related condition keys to the same
 condition block.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the condition key {{conditionKeyName}} can be overly
 permissive without also using the following condition keys: {{recommendedKeys}}.
 Condition keys like this one are more secure when paired with a related key. We
 recommend that you add the related condition keys to the same condition block."
```

Résolution de l'avertissement de sécurité

Certaines clés de condition sont plus sécurisées lorsqu'elles sont associées à d'autres clés de condition associées. AWS vous recommande d'inclure les clés de condition associées dans le même bloc de conditions que la clé de condition existante. Cela rend les autorisations accordées via la politique plus sécurisées.

Par exemple, vous pouvez utiliser la clé de condition `aws:VpcSourceIp` pour comparer l'adresse IP à partir de laquelle une demande a été effectuée avec l'adresse IP que vous spécifiez dans la politique. AWS vous recommande d'ajouter la clé de condition `aws:SourceVPC` associée. Cela vérifie si la demande provient du VPC que vous spécifiez dans la politique et de l'adresse IP que vous spécifiez.

Termes connexes

- clé de condition globale `aws:VpcSourceIp`
- clé de condition globale `aws:SourceVPC`
- Clés de condition globale
- Élément de condition
- Présentation des politiques JSON

Avertissement de sécurité : refuser avec clé de condition de balise non prise en charge pour le service

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Deny with unsupported tag condition key for service: Using the effect Deny with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes can be overly permissive: {{serviceNames}}. Actions for the listed services
 are not denied by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the effect Deny with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes can be
 overly permissive: {{serviceNames}}. Actions for the listed services are not denied
 by this statement. We recommend that you move these actions to a different statement
 without this condition key."
```

Résolution de l'avertissement de sécurité

L'utilisation de clés de condition de balise non prises en charge dans l'`Condition`élément d'une politique avec `"Effect": "Deny"` peut être trop permissive, car la condition est ignorée pour ce service. AWS recommande de supprimer les actions de service qui ne prennent pas en charge la clé de condition et de créer une autre instruction pour refuser l'accès à des ressources spécifiques pour ces actions.

Si vous utilisez la clé de condition `aws:ResourceTag` et qu'elle n'est pas prise en charge par une action de service, la clé n'est pas incluse dans le contexte de la demande. Dans ce cas, la condition

dans l'instruction Deny renvoie toujours `false` et l'action n'est jamais refusée. Cela se produit même si la ressource est correctement balisée.

Lorsqu'un service prend en charge la clé de condition `aws:ResourceTag`, vous pouvez utiliser des balises pour contrôler l'accès aux ressources de ce service. Ceci est connu sous le nom de contrôle d'accès basé sur les attributs (ABAC). Les services qui ne prennent pas en charge ces clés ont besoin que vous contrôliez l'accès aux ressources à l'aide du Contrôle d'accès basé sur les ressources (RBAC).

> **Note**
>
> Certains services autorisent la prise en charge de la clé de condition `aws:ResourceTag` pour un sous-ensemble de leurs ressources et actions. IAM Access Analyzer renvoie les résultats des actions de service qui ne sont pas prises en charge. Par exemple, Amazon S3 prend en charge `aws:ResourceTag` pour un sous-ensemble de ses ressources. Pour afficher tous les types de ressources disponibles dans Amazon S3 qui prennent en charge la clé de condition `aws:ResourceTag`, veuillez consulter Types de ressources définis par Amazon S3 dans la Référence des autorisations de service.

Par exemple, supposons que vous souhaitiez refuser l'accès pour supprimer les balises de ressources spécifiques qui sont balisées avec la paire clé-valeur `status=Confidential`. Supposons également que cela vous AWS Lambda permet de baliser et de débaliser des ressources, mais que cela ne prend pas en charge la clé de `aws:ResourceTag` condition. Pour refuser les actions de suppression pour AWS App Mesh et AWS Backup si cette balise est présente, utilisez la clé de `aws:ResourceTag` condition. Pour Lambda, utilisez une convention de dénomination des ressources incluant le préfixe `"Confidential"`. Incluez ensuite une instruction séparée qui empêche la suppression des ressources avec cette convention de dénomination.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyDeleteSupported",
            "Effect": "Deny",
            "Action": [
                "appmesh:DeleteMesh",
                "backup:DeleteBackupPlan"
                ],
```

```
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/status": "Confidential"
                }
            }
        },
        {
            "Sid": "DenyDeleteUnsupported",
            "Effect": "Deny",
            "Action": "lambda:DeleteFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:status-Confidential*"
        }
    ]
}
```

> **Warning**
>
> N'utilisez pas la IfExistsversion... de l'opérateur de condition pour contourner ce résultat. Cela signifie « Supprimer l'action si la clé est présente dans le contexte de la demande et que les valeurs correspondent. Sinon, refuser l'action. » Dans l'exemple précédent, le fait d'inclure l'action `lambda:DeleteFunction` dans l'instruction `DenyDeleteSupported` avec l'opérateur `StringEqualsIfExists` refuse toujours l'action. Pour cette action, la clé n'est pas présente dans le contexte et chaque tentative de suppression de ce type de ressource est refusée, que celle-ci soit ou non balisée.

Termes connexes

- Clés de condition globale
- Comparaison entre ABAC et RBAC
- Éléments de politique JSON IAM : Opérateurs de condition
- Élément de condition
- Présentation des politiques JSON

Avertissement de sécurité — Refuser NotAction avec étiquette non prise en charge, clé de condition pour le service

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Deny NotAction with unsupported tag condition key for service: Using the effect Deny
 with NotAction and the tag condition key {{conditionKeyName}} can be overly permissive
 because some service actions are not denied by this statement. This is because the
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the effect Deny with NotAction and the tag condition key
 {{conditionKeyName}} can be overly permissive because some service actions are not
 denied by this statement. This is because the condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction."
```

Résolution de l'avertissement de sécurité

L'utilisation de clés de condition de balise dans l'élément `Condition` d'une politique avec les éléments `NotAction` et `"Effect": "Deny"` peut être trop permissive. La condition est ignorée pour les actions de service qui ne prennent pas en charge la clé de condition. AWS recommande de réécrire la logique pour refuser une liste d'actions.

Si vous utilisez la clé de condition `aws:ResourceTag` avec `NotAction`, toutes les actions de service nouvelles ou existantes ne prenant pas en charge la clé ne sont pas refusées. AWS vous recommande de répertorier explicitement les actions que vous voulez refuser. IAM Access Analyzer renvoie un résultat distinct pour les actions répertoriées ne prenant pas en charge la clé de condition `aws:ResourceTag`. Pour plus d'informations, veuillez consulter Avertissement de sécurité : refuser avec clé de condition de balise non prise en charge pour le service.

Lorsqu'un service prend en charge la clé de condition `aws:ResourceTag`, vous pouvez utiliser des balises pour contrôler l'accès aux ressources de ce service. Ceci est connu sous le nom de contrôle d'accès basé sur les attributs (ABAC). Les services qui ne prennent pas en charge ces clés ont besoin que vous contrôliez l'accès aux ressources à l'aide du Contrôle d'accès basé sur les ressources (RBAC).

Termes connexes

- Clés de condition globale
- Comparaison entre ABAC et RBAC
- Éléments de politique JSON IAM : Opérateurs de condition

- Élément de condition
- Présentation des politiques JSON

Avertissement de sécurité : restreindre l'accès au principal de service

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Restrict access to service principal: Granting access to a service principal without
 specifying a source is overly permissive. Use aws:SourceArn, aws:SourceAccount,
 aws:SourceOrgID, or aws:SourceOrgPaths condition key to grant fine-grained access.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Granting access to a service principal without specifying a source
 is overly permissive. Use aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or
 aws:SourceOrgPaths condition key to grant fine-grained access."
```

Résolution de l'avertissement de sécurité

Vous pouvez spécifier Services AWS dans l'`Principal`élément d'une politique basée sur les ressources à l'aide d'un principal de service, qui est un identifiant du service. Lorsque vous accordez l'accès à un principal de service pour agir en votre nom, limitez l'accès. Vous pouvez éviter les politiques trop permissives en utilisant les touches`aws:SourceArn`, `aws:SourceAccount``aws:SourceOrgID`, ou de `aws:SourceOrgPaths` condition pour restreindre l'accès à une source spécifique, telle qu'un ARN Compte AWS, un identifiant d'organisation ou des chemins d'organisation spécifiques. La restriction de l'accès permet d'éviter un problème de sécurité appelé le Problème de l'adjoint confus.

Termes connexes

- Service AWS principes
- AWS clés de condition globales : aws : SourceAccount
- AWS clés de condition globales : aws : SourceArn
- AWS clés de condition globales : aws : SourceOrgId
- AWS clés de condition globales : aws : SourceOrgPaths
- Le problème de l'adjoint confus

Avertissement de sécurité : clé de condition pour oidc principal manquante

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing condition key for oidc principal: Using an Open ID Connect principal without a
 condition can be overly permissive. Add condition keys with a prefix that matches your
 federated OIDC principals to ensure that only the intended identity provider assumes
 the role.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using an Open ID Connect principal without a condition can be
 overly permissive. Add condition keys with a prefix that matches your federated OIDC
 principals to ensure that only the intended identity provider assumes the role."
```

Résolution de l'avertissement de sécurité

L'utilisation d'un principal Open ID Connect sans condition peut être trop permissive. Ajoutez des clés de condition avec un préfixe qui correspond à vos principaux OIDC fédérés pour garantir que seul le fournisseur d'identité prévu assume le rôle.

Termes connexes

- Création d'un rôle pour l'identité Web ou pour OpenID Connect Federation (console)

Avertissement de sécurité : clé de condition github repo manquante

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Missing github repo condition key: Granting a federated GitHub principal permissions
 without a condition key can allow more sources to assume the role than you intended.
 Add the token.actions.githubusercontent.com:sub condition key and specify the branch
 and repository name in the value.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Granting a federated GitHub principal permissions without a
 condition key can allow more sources to assume the role than you intended. Add the
```

```
token.actions.githubusercontent.com:sub condition key and specify the branch and
repository name in the value."
```

Résolution de l'avertissement de sécurité

Si vous l'utilisez GitHub en tant qu'IdP OIDC, la meilleure pratique consiste à limiter les entités qui peuvent assumer le rôle associé à l'IdP IAM. Lorsque vous incluez une `Condition` déclaration dans une politique de confiance des rôles, vous pouvez limiter le rôle à une GitHub organisation, un référentiel ou une branche spécifique. Vous pouvez utiliser la clé de condition `token.actions.githubusercontent.com:sub` pour limiter l'accès. Nous vous recommandons de limiter la condition à un ensemble spécifique de référentiels ou de branches. Si vous n'incluez pas cette condition, GitHub les actions provenant d'organisations ou de référentiels indépendants de votre volonté peuvent assumer les rôles associés à l' GitHub IdP IAM dans votre compte. AWS

Termes connexes

- Configuration d'un rôle pour le fournisseur d'identité GitHub OIDC

Suggestion : action de tableau vide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Empty array action: This statement includes no actions and does not affect the policy.
 Specify actions.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "This statement includes no actions and does not affect the policy.
 Specify actions."
```

Résolution de la suggestion

Les instructions doivent inclure un élément `Action` ou `NotAction` comprenant un ensemble d'actions. Lorsque l'élément est vide, l'instruction de politique ne fournit aucune autorisation. Spécifiez les actions dans l'élément `Action`.

- Éléments de politique IAM JSON : Action

Suggestion : condition de tableau vide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Empty array condition: There are no values for the condition key {{key}} and it does
 not affect the policy. Specify conditions.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "There are no values for the condition key {{key}} and it does not
 affect the policy. Specify conditions."
```

Résolution de la suggestion

La structure de l'élément `Condition` facultatif a besoin que vous utilisiez un opérateur de condition et une paire clé-valeur. Lorsque la valeur de la condition est vide, la condition renvoie `true` et l'instruction de politique ne fournit aucune autorisation. Spécifiez une valeur de condition.

- Éléments de politique JSON IAM : Condition

Suggestion — Condition de tableau vide ForAllValues

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Empty array condition ForAllValues: The ForAllValues prefix with an empty condition key
 matches only if the key {{key}} is missing from the request context. To determine if
 the request context is empty, we recommend that you use the Null condition operator
 with the value of true instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The ForAllValues prefix with an empty condition key matches only
 if the key {{key}} is missing from the request context. To determine if the request
 context is empty, we recommend that you use the Null condition operator with the value
 of true instead."
```

Résolution de la suggestion

La structure de l'élément `Condition` a besoin que vous utilisiez un opérateur de condition et une paire clé-valeur. L'opération d'ensemble `ForAllValues` teste si la valeur de chaque membre de la requête est un sous-ensemble de la clé de condition.

Lorsque vous utilisez `ForAllValues` avec une clé de condition vide, la condition ne correspond que s'il n'y a pas de clés dans la demande. AWS vous recommande plutôt d'utiliser l'opérateur de condition `Null` si vous voulez tester si un contexte de demande est vide.

- Clés de contexte à valeurs multiples
- Opérateur de condition null
- Éléments de politique JSON IAM : Condition

Suggestion — Condition de tableau vide ForAnyValue

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Empty array condition ForAnyValue: The ForAnyValue prefix with an empty condition key
 {{key}} never matches the request context and it does not affect the policy. Specify
 conditions.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The ForAnyValue prefix with an empty condition key {{key}} never
 matches the request context and it does not affect the policy. Specify conditions."
```

Résolution de la suggestion

La structure de l'élément `Condition` a besoin que vous utilisiez un opérateur de condition et une paire clé-valeur. L'opérateur d'ensemble `ForAnyValues` teste si au moins un membre de l'ensemble des valeurs de la demande correspond à au moins un membre de l'ensemble des valeurs de la clé de condition.

Lorsque vous utilisez `ForAnyValues` avec une clé de condition vide, la condition ne correspond jamais. Cela signifie que la déclaration n'a aucun effet sur la politique. AWS recommande de réécrire la condition.

- Clés de contexte à valeurs multiples
- Éléments de politique JSON IAM : Condition

Suggestion — Condition de tableau vide IfExists

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Empty array condition IfExists: The IfExists suffix with an empty condition key matches
 only if the key {{key}} is missing from the request context. To determine if the
 request context is empty, we recommend that you use the Null condition operator with
 the value of true instead.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The IfExists suffix with an empty condition key matches only if the
 key {{key}} is missing from the request context. To determine if the request context
 is empty, we recommend that you use the Null condition operator with the value of true
 instead."
```

Résolution de la suggestion

Le suffixe `...IfExists` modifie un opérateur de condition. Ceci signifie que si la clé de politique est présente dans le contexte de la demande, la clé doit être traitée comme spécifié dans la politique. Si la clé n'est pas présente, la condition évalue l'élément de condition comme vrai. »

Lorsque vous utilisez `...IfExists` avec une clé de condition vide, la condition ne correspond que s'il n'y a pas de clés dans la demande. AWS vous recommande plutôt d'utiliser l'opérateur de condition `Null` si vous voulez tester si un contexte de demande est vide.

- ... IfExists opérateurs de conditions
- Éléments de politique JSON IAM : Condition

Suggestion : principal de tableau vide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Empty array principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

Résolution de la suggestion

Vous devez utiliser l'élément `Principal` ou `NotPrincipal` dans les politiques de confiance pour les rôles IAM et dans les politiques basées sur les ressources. Les politiques basées sur les ressources sont des politiques que vous intégrez directement à une ressource.

Lorsque vous fournissez un tableau vide dans l'`Principal`élément d'une instruction, l'instruction n'a aucun effet sur la politique. AWS vous recommande de spécifier les principaux qui doivent avoir accès à la ressource.

- Éléments de politique JSON IAM : principal
- Éléments de politique JSON IAM : NotPrincipal

Suggestion : ressource de tableau vide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Empty array resource: This statement includes no resources and does not affect the
 policy. Specify resources.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "This statement includes no resources and does not affect the policy.
 Specify resources."
```

Résolution de la suggestion

Les instructions doivent inclure un élément `Resource` ou `NotResource`.

Lorsque vous fournissez un tableau vide dans l'élément ressource d'une instruction, l'instruction n'a aucun effet sur la politique. AWS vous recommande de spécifier les Amazon Resource Names (ARN) pour les ressources.

- Éléments de politique JSON IAM : Resource

- Éléments de politique JSON IAM : NotResource

Suggestion : condition d'objet vide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Empty object condition: This condition block is empty and it does not affect the
 policy. Specify conditions.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "This condition block is empty and it does not affect the policy.
 Specify conditions."
```

Résolution de la suggestion

La structure de l'élément `Condition` a besoin que vous utilisiez un opérateur de condition et une paire clé-valeur.

Lorsque vous fournissez un objet vide dans l'élément de condition d'une instruction, l'instruction n'a aucun effet sur la politique. Supprimez l'élément facultatif ou spécifiez des conditions.

- Éléments de politique JSON IAM : Condition

Suggestion : principal d'objet vide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Empty object principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

Résolution de la suggestion

Vous devez utiliser l'élément `Principal` ou `NotPrincipal` dans les politiques de confiance pour les rôles IAM et dans les politiques basées sur les ressources. Les politiques basées sur les ressources sont des politiques que vous intégrez directement à une ressource.

Lorsque vous fournissez un objet vide dans l'`Principal`élément d'une instruction, l'instruction n'a aucun effet sur la politique. AWS vous recommande de spécifier les principaux qui doivent avoir accès à la ressource.

- Éléments de politique JSON IAM : principal
- Éléments de politique JSON IAM : NotPrincipal

Suggestion : valeur sid vide

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Empty Sid value: Add a value to the empty string in the Sid element.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Add a value to the empty string in the Sid element."
```

Résolution de la suggestion

L'élément `Sid` (ID de instruction) facultatif est un identifiant que vous pouvez fournir pour l'instruction de politique. Vous pouvez affecter une valeur `Sid` à chaque instruction d'un tableau d'instructions. Si vous sélectionnez d'utiliser l'élément `Sid`, vous devez indiquer une valeur de chaîne.

Termes connexes

- Éléments de politique JSON IAM : Sid

Suggestion : améliorer une plage IP

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Improve IP range: The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}."
```

Résolution de la suggestion

Les conditions d'adresse IP doivent adopter le format CIDR standard, tel que 203.0.113.0/24 or 2001:DB8:1234:5678::/64. Lorsque vous incluez des bits non nuls après les bits masqués, ils ne sont pas pris en compte pour la condition. AWS vous recommande d'utiliser la nouvelle adresse figurant dans le message.

- Opérateurs de condition d'adresse IP
- Éléments de politique JSON IAM : Condition

Suggestion : Null avec qualificateur

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Null with qualifier: Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively."
```

Résolution de la suggestion

Dans l'élément `Condition`, vous créez des expressions dans lesquelles vous utilisez des opérateurs de condition tels que égal ou inférieur à pour comparer une condition de la politique avec des clés et des valeurs dans le contexte de la demande. Pour les demandes incluant plusieurs valeurs pour une seule clé de condition, vous devez utiliser les opérateurs définis `ForAllValues` ou `ForAnyValue`.

Lorsque vous utilisez l'opérateur de condition `Null` avec `ForAllValues`, l'instruction renvoie toujours `true`. Lorsque vous utilisez l'opérateur de `Null` condition with`ForAnyValue`, l'instruction

est toujours renvoyée`false`. AWS recommande d'utiliser l'opérateur de `StringLike` condition avec ces opérateurs définis.

Termes connexes

- Clés de contexte à valeurs multiples
- Opérateur de condition null
- Élément de condition

Suggestion : sous-ensemble d'adresses IP privées

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Private IP address subset: The values for condition key aws:SourceIp include a mix
 of private and public IP addresses. The private addresses will not have the desired
 effect. aws:SourceIp works only for public IP address ranges. To define permissions
 for private IP ranges, use aws:VpcSourceIp.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses will not have the desired effect.
 aws:SourceIp works only for public IP address ranges. To define permissions for
 private IP ranges, use aws:VpcSourceIp."
```

Résolution de la suggestion

La clé de condition globale `aws:SourceIp` fonctionne uniquement pour les plages d'adresses IP publiques.

Lorsque votre élément `Condition` inclut un mélange d'adresses IP privées et publiques, l'instruction peut ne pas avoir l'effet désiré. Vous pouvez spécifier des adresses IP privées avec `aws:VpcSourceIP`.

> **Note**
>
> La clé de condition globale `aws:VpcSourceIP` ne correspond que si la demande provient de l'adresse IP spécifiée et qu'elle passe par un point de terminaison de VPC.

- aws : clé de condition SourceIp globale
- aws : clé de condition VpcSourceIp globale
- Opérateurs de condition d'adresse IP
- Éléments de politique JSON IAM : Condition

Suggestion — NotIpAddress Sous-ensemble privé

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Private NotIpAddress subset: The values for condition key aws:SourceIp include a mix of
 private and public IP addresses. The private addresses have no effect. aws:SourceIp
 works only for public IP address ranges. To define permissions for private IP ranges,
 use aws:VpcSourceIp.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses have no effect. aws:SourceIp works
 only for public IP address ranges. To define permissions for private IP ranges, use
 aws:VpcSourceIp."
```

Résolution de la suggestion

La clé de condition globale `aws:SourceIp` fonctionne uniquement pour les plages d'adresses IP publiques.

Lorsque votre élément `Condition` inclut l'opérateur de condition `NotIpAddress` et un mélange d'adresses IP privées et publiques, l'instruction peut ne pas avoir l'effet désiré. Toutes les adresses IP publiques non spécifiées dans la politique correspondront. Aucune adresse IP privée ne correspondra. Pour obtenir cet effet, vous pouvez utiliser `NotIpAddress` avec `aws:VpcSourceIP` et spécifier les adresses IP privées qui ne doivent pas correspondre.

- aws : clé de condition SourceIp globale
- aws : clé de condition VpcSourceIp globale
- Opérateurs de condition d'adresse IP
- Éléments de politique JSON IAM : Condition

Suggestion : action redondante

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Redundant action: The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}."
```

Résolution de la suggestion

Lorsque vous utilisez des caractères génériques (*) dans l'`Action`élément, vous pouvez inclure des autorisations redondantes. AWS vous recommande de revoir votre politique et de n'inclure que les autorisations dont vous avez besoin. Cela peut vous aider à supprimer les actions redondantes.

Par exemple, les actions suivantes incluent l'action `iam:GetCredentialReport` deux fois.

```
"Action": [
        "iam:Get*",
        "iam:List*",
        "iam:GetCredentialReport"
    ],
```

Dans cet exemple, les autorisations sont définies pour chaque action IAM commençant par `Get` ou `List`. Lorsque IAM ajoute des opérations get ou list supplémentaires, cette politique les autorise. Vous pouvez vouloir autoriser toutes ces actions en lecture seule. L'action `iam:GetCredentialReport` est déjà incluse dans le cadre de `iam:Get*`. Pour supprimer les autorisations en double, vous pouvez supprimer `iam:GetCredentialReport`.

Vous recevez un résultat pour cette vérification de politique lorsque tout le contenu d'une action est redondant. Dans cet exemple, si l'élément incluait `iam:*CredentialReport`, il ne serait pas considéré comme redondant. En effet, il inclut `iam:GetCredentialReport`, qui est redondant, et `iam:GenerateCredentialReport`, qui ne l'est pas. La suppression de `iam:Get*` ou `iam:*CredentialReport` modifierait les autorisations de la politique.

- Éléments de politique JSON IAM : Action

AWS politiques gérées avec cette suggestion

AWS les politiques gérées vous permettent de démarrer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Les actions redondantes n'affectent pas les autorisations accordées par la politique. Lorsque vous utilisez une politique AWS gérée comme référence pour créer votre politique gérée par le client, il est AWS recommandé de supprimer les actions redondantes de votre politique.

Suggestion : valeur de condition redondante num

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Redundant condition value num: Multiple values in {{operator}} are redundant. Replace
 with the {{greatest/least}} single value for {{key}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Multiple values in {{operator}} are redundant. Replace with the
 {{greatest/least}} single value for {{key}}."
```

Résolution de la suggestion

Lorsque vous utilisez des opérateurs de condition numériques pour des valeurs similaires dans une clé de condition, vous pouvez créer un chevauchement qui entraîne des autorisations redondantes.

Par exemple, l'élément `Condition` suivant inclut plusieurs conditions `aws:MultiFactorAuthAge` dont les plages d'd'âge se chevauchent de 1 200 secondes.

```
"Condition": {
        "NumericLessThan": {
          "aws:MultiFactorAuthAge": [
            "2700",
            "3600"
          ]
        }
```

```
      }
```

Dans cet exemple, les autorisations sont définies si l'authentification multifacteur (MFA) a été effectuée il y a moins de 3 600 secondes (1 heure). Vous pouvez supprimer la valeur `2700` redondante.

- Opérateurs de condition numériques
- Éléments de politique JSON IAM : Condition

Suggestion : ressource redondante

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Redundant resource: The {{redundantResourceCount}} resource ARN(s) are redundant
 because they reference the same resource. Review the use of wildcards (*)
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The {{redundantResourceCount}} resource ARN(s) are redundant because
 they reference the same resource. Review the use of wildcards (*)"
```

Résolution de la suggestion

Lorsque vous utilisez des caractères génériques (*) dans les Amazon Resource Names (ARN), vous pouvez créer des autorisations de ressource redondante.

Par exemple, l'élément `Resource` suivant inclut plusieurs ARN avec des autorisations redondantes.

```
"Resource": [
            "arn:aws:iam::111122223333:role/jane-admin",
            "arn:aws:iam::111122223333:role/jane-s3only",
            "arn:aws:iam::111122223333:role/jane*"
        ],
```

Dans cet exemple, les autorisations sont définies pour tous les rôles dont le nom commence par `jane`. Vous pouvez supprimer les ARN `jane-admin` et `jane-s3only` redondants sans modifier les autorisations qui en résultent. Cela rend la politique dynamique. Elle définira les autorisations pour tous les futurs rôles commençant par `jane`. Si la politique vise à autoriser l'accès à un nombre

statique de rôles, supprimez le dernier ARN et répertoriez uniquement les ARN qui doivent être définis.

- Éléments de politique JSON IAM : Resource

AWS politiques gérées avec cette suggestion

AWS les politiques gérées vous permettent de démarrer AWS en attribuant des autorisations en fonction de cas AWS d'utilisation généraux.

Les ressources redondantes n'affectent pas les autorisations accordées par la politique. Lorsque vous utilisez une politique AWS gérée comme référence pour créer votre politique gérée par le client, il est AWS recommandé de supprimer les ressources redondantes de votre politique.

Suggestion : instruction redondante

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Redundant statement: The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement."
```

Résolution de la suggestion

L'élément `Statement` constitue l'élément principal d'une politique. Cet élément est obligatoire. L'élément `Statement` peut contenir une seule instruction ou un ensemble d'instructions individuelles.

Lorsque vous incluez la même instruction plusieurs fois dans une politique longue, les instructions sont redondantes. Vous pouvez supprimer l'une des instructions sans affecter les autorisations accordées par la politique. Lorsqu'une personne modifie une politique, elle peut modifier l'une des instructions sans mettre à jour son double. Cela peut entraîner un nombre d'autorisations plus élevé que prévu.

- Éléments de politique JSON IAM : Statement

Suggestion : caractère générique dans le nom du service

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Wildcard in service name: Avoid using wildcards (*, ?) in the service name because it
 might grant unintended access to other AWS services with similar names.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Avoid using wildcards (*, ?) in the service name because it might
 grant unintended access to other AWS services with similar names."
```

Résolution de la suggestion

Lorsque vous incluez le nom d'un AWS service dans une politique, il est AWS recommandé de ne pas inclure de caractères génériques (*, ?). Cela peut ajouter des autorisations non voulues à de futurs services. Par exemple, il existe plus d'une douzaine de AWS services dont le nom `*code*` contient le mot.

```
"Resource": "arn:aws:*code*::111122223333:*"
```

- Éléments de politique JSON IAM : Resource

Suggestion : autoriser avec la clé de condition d'étiquette non prise en charge pour le service

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Allow with unsupported tag condition key for service: Using the effect Allow with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes does not affect the policy: {{serviceNames}}. Actions for the listed service
 are not allowed by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the effect Allow with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes does not
```

```
affect the policy: {{serviceNames}}. Actions for the listed service are not allowed
by this statement. We recommend that you move these actions to a different statement
without this condition key."
```

Résolution de la suggestion

L'utilisation de clés de condition de balise non prises en charge dans l'`Condition`élément d'une politique avec n'`"Effect": "Allow"`affecte pas les autorisations accordées par la politique, car la condition est ignorée pour cette action de service. AWS recommande de supprimer les actions pour les services qui ne prennent pas en charge la clé de condition et de créer une autre instruction pour autoriser l'accès à des ressources spécifiques de ce service.

Si vous utilisez la clé de condition `aws:ResourceTag` et qu'elle n'est pas prise en charge par une action de service, la clé n'est pas incluse dans le contexte de la demande. Dans ce cas, la condition dans l'instruction `Allow` renvoie toujours `false` et l'action n'est jamais autorisée. Cela se produit même si la ressource est correctement balisée.

Lorsqu'un service prend en charge la clé de condition `aws:ResourceTag`, vous pouvez utiliser des balises pour contrôler l'accès aux ressources de ce service. Ceci est connu sous le nom de contrôle d'accès basé sur les attributs (ABAC). Les services qui ne prennent pas en charge ces clés ont besoin que vous contrôliez l'accès aux ressources à l'aide du Contrôle d'accès basé sur les ressources (RBAC).

> **Note**
>
> Certains services autorisent la prise en charge de la clé de condition `aws:ResourceTag` pour un sous-ensemble de leurs ressources et actions. IAM Access Analyzer renvoie les résultats des actions de service qui ne sont pas prises en charge. Par exemple, Amazon S3 prend en charge `aws:ResourceTag` pour un sous-ensemble de ses ressources. Pour afficher tous les types de ressources disponibles dans Amazon S3 qui prennent en charge la clé de condition `aws:ResourceTag`, veuillez consulter Types de ressources définis par Amazon S3 dans la Référence des autorisations de service.

Par exemple, supposons que vous vouliez autoriser des membres de l'équipe à afficher les détails des ressources spécifiques qui sont balisées avec la paire clé-valeur `team=BumbleBee`. Supposons également que cela vous AWS Lambda permet de baliser les ressources, mais que cela ne prend pas en charge la clé de `aws:ResourceTag` condition. Pour autoriser les actions d'affichage pour AWS App Mesh et AWS Backup si cette balise est présente, utilisez la clé de `aws:ResourceTag`

condition. Pour Lambda, utilisez une convention de dénomination des ressources incluant le nom de l'équipe comme préfixe. Incluez ensuite une instruction séparée qui permet l'affichage des ressources avec cette convention de dénomination.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewSupported",
            "Effect": "Allow",
            "Action": [
                "appmesh:DescribeMesh",
                "backup:GetBackupPlan"
                ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/team": "BumbleBee"
                }
            }
        },
        {
            "Sid": "AllowViewUnsupported",
            "Effect": "Allow",
            "Action": "lambda:GetFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:team-BumbleBee*"
        }
    ]
}
```

> **Warning**
>
> N'utilisez pas la version de l'opérateur de condition `Not` avec `"Effect": "Allow"` pour contourner ce résultat. Ces opérateurs de condition fournissent une correspondance annulée. Cela signifie qu'une fois la condition évaluée, le résultat est annulé. Dans l'exemple précédent, le fait d'inclure l'action `lambda:GetFunction` dans l'instruction `AllowViewSupported` avec l'opérateur `StringNotEquals` autorise toujours l'action, que la ressource soit balisée ou non.
>
> N'utilisez pas la IfExistsversion... de l'opérateur de condition pour contourner ce résultat. Cela signifie « Autoriser l'action si la clé est présente dans le contexte de la demande et que les valeurs correspondent. Sinon, autoriser l'action. » Dans l'exemple précédent, le fait

d'inclure l'action `lambda:GetFunction` dans l'instruction `AllowViewSupported` avec l'opérateur `StringEqualsIfExists` autorise toujours l'action. Pour cette action, la clé n'est pas présente dans le contexte et chaque tentative d'affichage de ce type de ressource est autorisée, que la ressource soit ou non étiquetée.

Termes connexes

- Clés de condition globale
- Éléments de politique JSON IAM : Opérateurs de condition
- Élément de condition
- Présentation des politiques JSON

Suggestion — Autoriser NotAction avec étiquette non prise en charge, clé de condition pour le service

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Allow NotAction with unsupported tag condition key for service: Using the effect Allow
 with NotAction and the tag condition key {{conditionKeyName}} allows only service
 actions that support the condition key. The condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "Using the effect Allow with NotAction and the tag condition key
 {{conditionKeyName}} allows only service actions that support the condition key. The
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction."
```

Résolution de la suggestion

L'utilisation de clés de condition de balise non prises en charge dans l'élément `Condition` d'une politique avec l'élément `NotAction` et `"Effect": "Allow"` n'affecte pas les autorisations accordées par la politique. La condition est ignorée pour les actions de service qui ne prennent pas en charge la clé de condition. AWS recommande de réécrire la logique pour autoriser une liste d'actions.

Si vous utilisez la clé de condition `aws:ResourceTag` avec `NotAction`, toutes les actions de service nouvelles ou existantes ne prenant pas en charge la clé ne sont pas autorisées. AWS vous recommande de répertorier explicitement les actions que vous voulez autoriser. IAM Access Analyzer renvoie un résultat distinct pour les actions répertoriées ne prenant pas en charge la clé de condition `aws:ResourceTag`. Pour plus d'informations, veuillez consulter Suggestion : autoriser avec la clé de condition d'étiquette non prise en charge pour le service.

Lorsqu'un service prend en charge la clé de condition `aws:ResourceTag`, vous pouvez utiliser des balises pour contrôler l'accès aux ressources de ce service. Ceci est connu sous le nom de contrôle d'accès basé sur les attributs (ABAC). Les services qui ne prennent pas en charge ces clés ont besoin que vous contrôliez l'accès aux ressources à l'aide du Contrôle d'accès basé sur les ressources (RBAC).

Termes connexes

- Clés de condition globale
- Comparaison entre ABAC et RBAC
- Éléments de politique JSON IAM : Opérateurs de condition
- Élément de condition
- Présentation des politiques JSON

Suggestion : clé de condition recommandée pour le principal de service

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Recommended condition key for service principal: To restrict access to the service
 principal {{servicePrincipalPrefix}} operating on your behalf, we recommend
 aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of
 {{key}}.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "To restrict access to the service principal
 {{servicePrincipalPrefix}} operating on your behalf, we recommend aws:SourceArn,
 aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of {{key}}."
```

Résolution de la suggestion

Vous pouvez spécifier Services AWS dans l'`Principal`élément d'une politique basée sur les ressources à l'aide d'un principal de service, qui est un identifiant du service. Vous devez utiliser les clés de condition `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` ou `aws:SourceOrgPaths` lors de l'octroi d'un accès à des principaux de service plutôt qu'à d'autres clés de condition, telles que `aws:Referer`. Cela vous aide à prévenir un problème de sécurité appelé le problème de l'adjoint confus.

Termes connexes

- Service AWS principes
- AWS clés de condition globales : aws : SourceAccount
- AWS clés de condition globales : aws : SourceArn
- AWS clés de condition globales : aws : SourceOrgId
- AWS clés de condition globales : aws : SourceOrgPaths
- Le problème de l'adjoint confus

Suggestion : clé de condition non pertinente dans la politique

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Irrelevant condition key in policy: The condition key {{condition-key}} is not relevant
 for the {{resource-type}} policy.  Use this key in an identity-based policy to govern
 access to this resource.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The condition key {{condition-key}} is not relevant for the
 {{resource-type}} policy.  Use this key in an identity-based policy to govern access
 to this resource."
```

Résolution de la suggestion

Certaines clés de condition ne sont pas pertinentes pour les politiques basées sur les ressources. Par exemple, la clé de condition `s3:ResourceAccount` n'est pas pertinente pour la politique basée sur les ressources attachée à un compartiment Amazon S3 ou à un type de ressource de point d'accès Amazon S3.

Vous devez utiliser la clé de condition dans une politique basée sur l'identité pour contrôler l'accès à la ressource.

Termes connexes

- Politiques basées sur l'identité et politiques basées sur les ressources

Suggestion : principal redondant dans la politique d'approbation de rôle

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Redundant principal in role trust policy: The assumed-role principal
 {{redundant_principal}} is redundant with its parent role {{parent_role}}. Remove the
 assumed-role principal.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The assumed-role principal {{redundant_principal}} is redundant with
 its parent role {{parent_role}}. Remove the assumed-role principal."
```

Résolution de la suggestion

Si vous spécifiez à la fois un principal avec rôle assumé et son rôle parent dans l'élément `Principal` d'une politique, celle-ci n'autorise ou ne refuse aucune autorisation différente. Par exemple, il est redondant si vous spécifiez l'élément `Principal` au format suivant :

```
"Principal": {
            "AWS": [
            "arn:aws:iam::AWS-account-ID:role/rolename",
            "arn:aws:iam::AWS-account-ID:assumed-role/rolename/rolesessionname"
        ]
```

Nous recommandons de supprimer le principal avec rôle assumé.

Termes connexes

- Principaux de séance de rôle

Suggestion : confirmer le type de réclamation de l'audience

Dans le AWS Management Console, le résultat de cette vérification inclut le message suivant :

```
Confirm audience claim type: The 'aud' (audience) claim key identifies the recipients
 that the JSON web token is intended for. Audience claims can be multivalued or single-
valued. If the claim is multivalued, use a ForAllValues or ForAnyValue qualifier. If
 the claim is single-valued, do not use a qualifier.
```

Dans les appels programmatiques à l' AWS API AWS CLI or, le résultat de cette vérification inclut le message suivant :

```
"findingDetails": "The 'aud' (audience) claim key identifies the recipients that the
 JSON web token is intended for. Audience claims can be multivalued or single-valued.
 If the claim is multivalued, use a ForAllValues or ForAnyValue qualifier. If the claim
 is single-valued, do not use a qualifier."
```

Résolution de la suggestion

La clé de réclamation (d'audience) `aud` est un identifiant unique pour votre application qui vous est délivré lorsque vous enregistrez votre application auprès de l'IdP, et elle identifie les destinataires auxquels le jeton web JSON est destiné. Les reclamations d'audience peuvent être à valeurs multiples ou à valeur unique. Si la réclamation est à valeurs multiples, utilisez un opérateur d'ensemble de conditions `ForAllValues` ou `ForAnyValue`. Si la réclamation est à valeur unique, n'utilisez pas d'opérateur d'ensemble de conditions.

Termes connexes

- Création d'un rôle pour l'identité Web ou pour OpenID Connect Federation (console)
- Clés de contexte à valeurs multiples
- Clés de condition à valeur unique ou à valeurs multiples

Vérifications de politiques personnalisées de l’analyseur d’accès IAM

Vous pouvez valider vos politiques par rapport aux normes de sécurité spécifiées à l’aide de vérifications de politique personnalisées AWS Identity and Access Management Access Analyzer . Vous pouvez exécuter les types de contrôles de politique personnalisés suivants :

- Vérifier par rapport à une stratégie de référence : Lorsque vous modifiez une politique, vous pouvez vérifier si celle mise à jour accorde un nouvel accès par rapport à une de référence, telle qu'une version existante de cette politique. Vous pouvez exécuter cette vérification lorsque vous modifiez une politique à l'aide de AWS Command Line Interface (AWS CLI), de l'API IAM Access Analyzer (API) ou de l'éditeur de stratégie JSON dans la console IAM.
- Vérifiez par rapport à une liste d'actions ou de ressources IAM : vous pouvez vérifier que des actions ou des ressources IAM spécifiques ne sont pas autorisées par votre politique. Si seules des actions sont spécifiées, IAM Access Analyzer vérifie l'accès aux actions sur toutes les ressources de la politique. Si seules les ressources sont spécifiées, IAM Access Analyzer vérifie quelles actions ont accès aux ressources spécifiées. Si des actions et des ressources sont spécifiées, IAM Access Analyzer vérifie lesquelles des actions spécifiées ont accès aux ressources spécifiées. Vous pouvez exécuter cette vérification lorsque vous créez ou modifiez une politique à l'aide d' AWS CLI ou de l'API.
- Vérifier l'accès public : vous pouvez vérifier si une politique de ressources peut accorder un accès public à un type de ressource spécifique. Vous pouvez exécuter cette vérification lorsque vous créez ou modifiez une politique à l'aide de l'API AWS CLI ou de l'API. Ce type de vérification des politiques personnalisées diffère de la prévisualisation de l'accès car il ne nécessite aucun compte ni aucun contexte d'analyseur d'accès externe. Les aperçus d'accès vous permettent de prévisualiser les résultats d'IAM Access Analyzer avant de déployer les autorisations relatives aux ressources, tandis que le contrôle personnalisé détermine si un accès public peut être accordé par une politique.

Des frais sont associés à chaque vérification de politique personnalisée. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

Comment fonctionnent les vérifications de politique personnalisées

Vous pouvez exécuter des vérifications de politique personnalisées sur les politiques basées sur l'identité et les ressources. Les vérifications de politique personnalisées ne reposent pas sur des techniques de correspondance de modèles ou sur l'examen des journaux d'accès pour déterminer si un nouvel accès ou spécifique est autorisé par une politique. À l'instar des résultats des accès externes, les vérifications de politique personnalisées sont basées sur Zelkova. Zelkova traduit les politiques IAM en déclarations logiques équivalentes, et exécute une suite de solveurs logiques polyvalents et spécialisés (théories du module de satisfiabilité) contre le problème. Pour vérifier l'existence d'un accès nouveau ou spécifié, l'analyseur d'accès IAM applique Zelkova de manière répétée à une politique. Les requêtes deviennent de plus en plus spécifiques pour caractériser les

classes de comportements que la politique autorise sur la base du contenu de la politique. Pour en savoir plus sur les théories du module de satisfiabilité, consultez Théories du module de satisfiabilité.

Dans de rares cas, l'analyseur d'accès IAM n'est pas en mesure de déterminer complètement si une déclaration de politique octroie un accès nouveau ou spécifique. Dans ces cas, il commet une erreur en déclarant un faux positif en échouant à la vérification de politique personnalisées. L'analyseur d'accès IAM est conçu pour fournir une évaluation exhaustive des politiques et s'efforce de réduire les faux négatifs. Cette approche signifie que l'analyseur d'accès IAM offre un degré élevé d'assurance qu'une vérification réussie signifie que l'accès n'a pas été accordé par la politique. Vous pouvez inspecter les vérifications qui ont échoué manuellement en consultant la déclaration de politique indiquée dans la réponse de l'analyseur d'accès IAM.

Exemples de politiques de référence pour vérifier les nouveaux accès

Vous pouvez trouver des exemples de politiques de référence et apprendre à configurer et exécuter une vérification de politique personnalisée pour les nouveaux accès dans le référentiel d'exemples de politiques personnalisées d'IAM Access Analyzer sur GitHub.

Avant d'utiliser ces exemples

Avant d'utiliser ces exemples de politiques de référence, effectuez les opérations suivantes :

- Vérifiez attentivement et personnalisez les politiques de référence en fonction de vos exigences uniques.
- Testez soigneusement les politiques de référence dans votre environnement avec les Services AWS que vous utilisez.

  Les politiques de référence illustrent la mise en œuvre et l'utilisation de vérifications de politique personnalisées. Ils ne sont pas destinés à être interprétés comme des recommandations ou des bonnes pratiques AWS officielles à mettre en œuvre exactement comme indiqué. Il est de votre responsabilité de tester soigneusement les politques de référence afin de déterminer si elles conviennent pour répondre aux exigences de sécurité de votre environnement.
- Les vérifications de politique personnalisées sont indépendantes de l'environnement dans leur analyse. Leur analyse ne prend en compte que les informations contenues dans les politiques d'entrée. Par exemple, les vérifications de politiques personnalisées ne permettent pas de vérifier si un compte est membre d'une AWS organisation spécifique. Par conséquent, les vérifications de politique personnalisées ne peuvent pas comparer les

nouveaux accès en fonction des valeurs des clés de condition pour les clés de condition `aws:PrincipalOrgId` et `aws:PrincipalAccount`.

Inspection des vérifications de politiques personnalisées ayant échoué

Lorsqu'une vérification de politique personnalisée échoue, la réponse de l'analyseur d'accès IAM inclut l'ID d'instruction (`Sid`) de celle de politique responsable de l'échec de la vérification. Bien que l'ID de déclaration soit un élément de politique facultatif, nous vous recommandons d'ajouter un ID de déclaration pour chaque déclaration de politique. La vérification de politique personnalisée renvoie également un index de déclaration pour aider à identifier la raison de l'échec de la vérification. L'index de déclaration suit une numérotation à base de zéro, où la première déclaration est référencée par 0. Lorsque plusieurs déclarations entraînent l'échec d'une vérification, celle-ci ne renvoie qu'un seul ID de déclaration à la fois. Nous vous recommandons de corriger la déclaration mise en évidence dans le motif et de réexécuter la vérification jusqu'à ce qu'elle soit validée.

Validation des politiques à l'aide de vérifications de politique personnalisées (console)

En option, vous pouvez exécuter une vérification de politique personnalisées lorsque vous modifiez une politique dans l'éditeur de politique JSON dans la console IAM. Vous pouvez vérifier si la politique mise à jour accorde un nouvel accès par rapport à la version existante.

Pour vérifier les nouveaux accès lors de la modification des politiques IAM JSON

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, sélectionnez Policies (Politiques).
3. Dans la liste des politiques, choisissez le nom de la politique à modifier. Vous pouvez utiliser la zone de recherche pour filtrer la liste des politiques.
4. Choisissez l'onglet Autorisations, puis Modifier.
5. Choisissez l'option JSON et mettez à jour votre politique.
6. Dans le panneau de validation de la politique ci-dessous, sélectionnez l'onglet Vérifier un nouvel accès, puis sélectionnez Vérifier la politique. Si les autorisations modifiées accordent un nouvel accès, la déclaration sera mise en évidence dans le volet de validation de la politique.
7. Si vous n'avez pas l'intention d'accorder un nouvel accès, mettez à jour la déclaration de politique et choisissez Vérifier la politique jusqu'à ce qu'aucun nouvel accès ne soit détecté.

> **Note**
>
> Des frais sont associés à chaque vérification pour un nouvel accès. Pour plus d'informations sur les tarifs, consultez la Tarification de l'analyseur d'accès IAM.

8. Choisissez Suivant.
9. Sur la page examiner et enregistrer, examinez les Autorisations définies dans cette politique, puis choisissez Enregistrer les modifications.

Validation des politiques à l'aide de vérifications de politiques personnalisées (AWS CLI ou API)

Vous pouvez exécuter des vérifications de politique personnalisées par IAM Access Analyzer à partir de l'API AWS CLI ou de l'API d'IAM Access Analyzer.

Pour exécuter les vérifications de politiques personnalisées de l'analyseur d'accès IAM (AWS CLI)

- Pour vérifier si un nouvel accès est autorisé pour une politique mise à jour par rapport à la politique existante, exécutez la commande suivante : check-no-new-access
- Pour vérifier si l'accès spécifié n'est pas autorisé par une politique, exécutez la commande suivante : check-access-not-granted
- Pour vérifier si une politique de ressources peut accorder un accès public à un type de ressource spécifié, exécutez la commande suivante : check-no-public-access

Pour exécuter les vérifications de politiques personnalisées de l'analyseur d'accès IAM (API)

- Pour vérifier si un nouvel accès est autorisé pour une politique mise à jour par rapport à la politique existante, utilisez l'opération d'API CheckNoNewAccess.
- Pour vérifier si l'accès spécifié n'est pas autorisé par une politique, utilisez l'opération d'API CheckAccessNotGranted.
- Pour vérifier si une politique de ressources peut accorder un accès public à un type de ressource spécifique, utilisez l'opération CheckNoPublicAccessAPI.

Génération d'une politique IAM Access Analyzer

En tant qu'administrateur ou développeur, vous pouvez octroyer plus d'autorisations à des entités IAM (utilisateurs ou rôles) qu'elle n'en ont besoin. IAM propose plusieurs options pour vous aider à affiner les autorisations que vous octroyez. Une option consiste à générer une politique IAM basée sur une activité d'accès pour une entité. IAM Access Analyzer examine vos AWS CloudTrail journaux et génère un modèle de politique qui contient les autorisations utilisées par l'entité dans la plage de dates spécifiée. Vous pouvez utiliser le modèle pour créer une politique avec des autorisations précises qui accordent uniquement les autorisations requises pour prendre en charge votre cas d'utilisation spécifique.

Rubriques

- Processus de génération de politique
- Informations de niveau service et action
- Points à noter sur la génération de politiques
- Autorisations nécessaires pour générer une politique
- Génération d'une politique basée sur CloudTrail l'activité (console)
- Générer une politique en utilisant AWS CloudTrail les données d'un autre compte
- Génération d'une politique basée sur CloudTrail l'activité (AWS CLI)
- Générer une politique basée sur CloudTrail l'activité (AWS API)
- Services de génération d'une politique d'Analyseur d'accès IAM

Processus de génération de politique

IAM Access Analyzer analyse vos CloudTrail événements pour identifier les actions et les services utilisés par une entité IAM (utilisateur ou rôle). Il génère ensuite une politique IAM basée sur cette activité. Vous pouvez ajuster les autorisations d'une entité en remplaçant la politique d'autorisations étendues attachée à l'entité par la politique générée. Voici un aperçu général du processus de génération de politique.

- Configuration pour la génération de modèles de politique : vous spécifiez une période maximale de 90 jours pour qu'IAM Access Analyzer analyse vos événements historiques AWS CloudTrail . Vous devez spécifier un rôle de service existant ou en créer un nouveau. Le rôle de service permet à IAM Access Analyzer d'accéder à votre historique et CloudTrail aux dernières informations auxquelles vous avez accédé afin d'identifier les services et les actions utilisés. Vous devez

spécifier la CloudTrail trace qui enregistre les événements pour le compte avant de pouvoir générer une politique. Pour plus d'informations sur les quotas de CloudTrail données d'IAM Access Analyzer, consultez la section Quotas d'IAM Access Analyzer.

- Générer une politique — IAM Access Analyzer génère une politique basée sur l'activité d'accès associée à vos CloudTrail événements.
- Examiner et personnaliser la politique : Une fois la politique générée, vous pouvez consulter les services et actions qui ont été utilisés par l'entité pendant la plage de dates spécifiée. Vous pouvez personnaliser davantage la politique en ajoutant ou en supprimant des autorisations, en spécifiant des ressources et en ajoutant des conditions au modèle de politique.
- Créer et attacher une politique : Vous avez la possibilité d'enregistrer la politique générée en créant une politique gérée. Vous pouvez attacher la politique que vous créez à l'utilisateur ou au rôle dont l'activité a été utilisée pour générer la politique.

Informations de niveau service et action

Lorsque IAM Access Analyzer génère une politique IAM, des informations sont renvoyées pour vous aider à la personnaliser. Deux catégories d'informations peuvent être renvoyées lorsqu'une politique est générée :

- Politique avec informations au niveau de l'action — Pour certains AWS services, tels qu'Amazon EC2, IAM Access Analyzer peut identifier les actions trouvées dans CloudTrail vos événements et répertorie les actions utilisées dans la politique qu'il génère. Pour obtenir la liste des services pris en charge, consultez Services de génération d'une politique d'Analyseur d'accès IAM. Pour certains services, IAM Access Analyzer vous invite à ajouter des actions pour les services à la politique générée.
- Politique avec des informations de niveau service : IAM Access Analyzer utilise les informations relatives aux derniers services consultés pour créer un modèle de politique avec tous les services récemment utilisés. Lorsque vous utilisez le AWS Management Console, nous vous invitons à consulter les services et à ajouter des actions pour compléter la politique.

Pour obtenir la liste des actions de chaque service, consultez la section Actions, ressources et clés de condition pour les AWS services dans la référence d'autorisation des services.

Points à noter sur la génération de politiques

Avant de générer une politique, veuillez consulter les conseils clés suivants.

- Activer un CloudTrail suivi : vous devez avoir activé un CloudTrail suivi pour que votre compte puisse générer une politique basée sur l'activité d'accès. Lorsque vous créez un CloudTrail suivi, les CloudTrail événements liés à celui-ci sont envoyés vers un compartiment Amazon S3 que vous spécifiez. Pour savoir comment créer un CloudTrail parcours, consultez la section Création d'un parcours pour votre AWS compte dans le Guide de AWS CloudTrail l'utilisateur.
- Événements de données non disponibles : l'analyseur d'accès IAM n'identifie pas l'activité au niveau action pour les événements de données, tels que les événements de données Amazon S3, dans les politiques générées.
- PassRole— L'`iam:PassRole`action n'est pas suivie CloudTrail et n'est pas incluse dans les politiques générées.
- Réduire le temps de génération des politiques– Pour générer une politique plus rapidement, réduisez la plage de dates spécifiée lors de la configuration.
- Utilisation CloudTrail à des fins d'audit : n'utilisez pas la génération de politiques à des fins d'audit ; CloudTrail utilisez-la plutôt. Pour plus d'informations sur l'utilisation CloudTrail, consultez la section Journalisation des appels IAM et AWS STS API avec AWS CloudTrail.
- Actions refusées : la génération de politiques passe en revue tous les CloudTrail événements, y compris les actions refusées.
- Une politique par console IAM : Vous pouvez générer une politique à la fois dans la console IAM.
- Disponibilité des politiques générées dans la console IAM : Vous pouvez consulter une politique générée dans la console IAM pendant 7 jours au maximum après sa génération. Après 7 jours, vous devez générer une nouvelle politique.
- Quotas de génération de politiques : pour plus d'informations sur les quotas de génération de politiques d'analyseur d'accès IAM, veuillez consulter Quotas IAM Access Analyzer.
- Les tarifs standard d'Amazon S3 s'appliquent : lorsque vous utilisez la fonctionnalité de génération de politiques, IAM Access Analyzer examine CloudTrail les journaux de votre compartiment S3. Aucun frais de stockage supplémentaire n'est facturé pour accéder à vos CloudTrail journaux afin de générer des politiques. AWS facture les tarifs standard d'Amazon S3 pour les demandes et le transfert de données des CloudTrail journaux stockés dans votre compartiment S3.
- AWS Control Tower support — La génération de politiques ne prend pas en AWS Control Tower charge la génération de politiques.

Autorisations nécessaires pour générer une politique

Les autorisations dont vous avez besoin pour générer une politique pour la première fois diffèrent de celles dont vous avez besoin pour générer une politique les fois suivantes.

Première configuration

Lorsque vous générez une politique pour la première fois, vous devez choisir un rôle de service existant approprié dans votre compte ou créer un nouveau rôle de service. Le rôle de service permet à IAM Access Analyzer d'accéder aux informations de votre CloudTrail compte auxquelles vous avez accédé pour la dernière fois. Seuls les administrateurs doivent disposer des autorisations nécessaires pour créer et configurer des rôles. Par conséquent, nous recommandons qu'un administrateur crée le rôle de service lors de la première installation. Pour en savoir plus sur les autorisations requises pour créer des rôles de service, voir Création d'un rôle pour déléguer des autorisations à un AWS service.

Autorisations nécessaires pour le rôle de service

Lorsque vous créez un rôle de service, vous configurez deux politiques pour ce rôle. Vous attachez une politique d'autorisations IAM au rôle qui spécifie ce que le rôle peut faire. Vous lui attachez également une politique d'approbation de rôle, qui spécifie le principal qui peut utiliser le rôle.

Le premier exemple de politique correspond à une politique d'autorisations à attacher au rôle de service requis pour générer une politique. Le deuxième exemple de politique correspond à une politique d'approbation de rôle à attacher au rôle de service. Vous pouvez utiliser ces politiques pour créer un rôle de service lorsque vous utilisez l' AWS API ou AWS CLI pour générer une politique. Lorsque vous utilisez la console IAM pour créer un rôle de service dans le cadre du processus de génération de politique, nous générons ces politiques pour vous.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "cloudtrail:GetTrail",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetServiceLastAccessedDetails",
```

```
                "iam:GenerateServiceLastAccessedDetails"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
        }
    ]
}
```

L'exemple de politique suivant correspond à une politique d'approbation de rôle avec les autorisations permettant à IAM Access Analyzer d'endosser le rôle.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "access-analyzer.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Utilisations ultérieures

Pour générer des politiques dans le AWS Management Console, un utilisateur IAM doit disposer d'une politique d'autorisations lui permettant de transmettre le rôle de service utilisé pour la génération des politiques à IAM Access Analyzer. `iam:PassRole`est généralement accompagné de `iam:GetRole` manière à ce que l'utilisateur puisse obtenir les détails du rôle à transmettre. Dans cet exemple, l'utilisateur peut transférer uniquement les rôles qui existent dans le compte spécifié avec des noms qui commencent par `AccessAnalyzerMonitorServiceRole*`. Pour en savoir

plus sur le transfert de rôles IAM à des AWS services, consultez la section Accorder à un utilisateur l'autorisation de transférer un rôle à un AWS service.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowUserToPassRole",
      "Effect": "Allow",
      "Action": [
        "iam:GetRole",
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::123456789012:role/service-role/
AccessAnalyzerMonitorServiceRole*"
    }
  ]
}
```

Vous devez également disposer des autorisations IAM Access Analyzer suivantes pour générer des politiques dans l' AWS Management Console AWS API ou AWS CLI comme indiqué dans la déclaration de politique suivante.

```
{
  "Sid": "AllowUserToGeneratePolicy",
  "Effect": "Allow",
  "Action": [
    "access-analyzer:CancelPolicyGeneration",
    "access-analyzer:GetGeneratedPolicy",
    "access-analyzer:ListPolicyGenerations",
    "access-analyzer:StartPolicyGeneration"
  ],
  "Resource": "*"
}
```

Pour la première fois et les utilisations ultérieures

Lorsque vous utilisez le AWS Management Console pour générer une politique, vous devez être `cloudtrail:ListTrails` autorisé à répertorier les CloudTrail sentiers de votre compte, comme indiqué dans la déclaration de politique suivante.

```
{
```

```
  "Sid": "AllowUserToListTrails",
  "Effect": "Allow",
  "Action": [
    "CloudTrail:ListTrails"
  ],
  "Resource": "*"
}
```

Génération d'une politique basée sur CloudTrail l'activité (console)

Vous pouvez générer une politique pour un utilisateur ou un rôle IAM.

Étape 1 : générer une politique basée sur CloudTrail l'activité

La procédure suivante explique comment générer une politique pour un rôle à l'aide de la AWS Management Console.

Générer une politique pour un rôle IAM

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de gauche, sélectionnez Roles (Rôles).

   > **Note**
   >
   > Les étapes à suivre pour générer une politique basée sur l'activité d'un utilisateur IAM sont presque identiques. Dans ce cas, sélectionnez Users (Utilisateurs) au lieu de Roles (Rôles).

3. Dans la liste des rôles de votre compte, sélectionnez le nom du rôle dont vous souhaitez utiliser l'activité pour générer une politique.
4. Dans l'onglet Autorisations, dans la section Générer une politique basée sur CloudTrail les événements, choisissez Générer une politique.
5. Sur la page Générer une politique, spécifiez la période pendant laquelle vous souhaitez qu'IAM Access Analyzer analyse vos CloudTrail événements pour déterminer les actions entreprises avec le rôle. Vous pouvez choisir une plage allant jusqu'à 90 jours. Nous vous recommandons de choisir la période la plus courte possible afin de réduire le temps de génération de politique.
6. Dans la section CloudTrail Accès, choisissez un rôle existant approprié ou créez-en un nouveau s'il n'existe pas de rôle approprié. Le rôle donne à IAM Access Analyzer l'autorisation d'accéder

à vos CloudTrail données en votre nom afin d'examiner les activités d'accès afin d'identifier les services et les actions qui ont été utilisés. Pour en savoir plus sur les autorisations requises pour ce rôle, veuillez consulter Autorisations nécessaires pour générer une politique.

7. Dans la CloudTrail section Suivi à analyser, spécifiez le CloudTrail suivi qui enregistre les événements du compte.

   Si vous choisissez un CloudTrail parcours qui stocke les journaux sur un autre compte, une boîte d'information concernant l'accès entre comptes s'affiche. L'accès intercompte nécessite une configuration supplémentaire. Pour en savoir plus, veuillez consulter Choose a role for cross-account access plus loin dans cette rubrique.
8. Sélectionnez Generate policy (Générer une politique).
9. Lorsque la génération de politique est en cours, vous revenez à la page Roles Summary (Récapitulatif des rôles) sous l'onglet Permissions (Autorisations). Attendez que le statut de la section Policy request details (Détails de la demande de politique) affiche Success (Succès), puis sélectionnez View generated policy (Afficher la politique générée). Vous pouvez afficher la politique générée pendant sept jours au maximum. Si vous générez une autre politique, la politique existante est remplacée par la nouvelle politique que vous générez.

Étape 2 : Examiner les autorisations et ajouter des actions pour les services utilisés

Examinez les services et les mesures que l'analyseur d'accès IAM a constaté être utilisés par le rôle. Vous pouvez ajouter des actions pour tous les services qui ont été utilisés dans le modèle de politique généré.

1. Examinez les sections suivantes :

   - Sur la page Review permissions (Vérifier les autorisations), veuillez consulter la liste des Actions included in the generated policy (Actions incluses dans la politique générée). La liste affiche les services et actions que l'analyseur d'accès IAM a constaté être utilisés par le rôle dans la plage de dates spécifiée.
   - La section Services used (Services utilisés) affiche les services supplémentaires que l'analyseur d'accès IAM a constaté être utilisés par le rôle dans la plage de dates spécifiée. Il est possible que les informations sur les actions utilisées ne soient pas disponibles pour les services répertoriés dans cette section. Utilisez les menus de chaque service répertorié pour choisir manuellement les actions à inclure dans la politique.
2. Lorsque vous avez terminé d'ajouter des actions, cliquez sur Next (Suivant).

Étape 3 : Personnaliser la politique générée

Vous pouvez personnaliser la politique en ajoutant ou en supprimant des autorisations ou en spécifiant des ressources.

Pour personnaliser la politique générée

1. Mettez à jour le modèle de politique. Le modèle de politique contient des espaces réservés ARN de ressource pour les actions qui prennent en charge les autorisations au niveau des ressources, comme illustré dans l'image suivante. Les autorisations au niveau des ressources font référence à la possibilité de spécifier les ressources sur lesquelles les utilisateurs sont autorisés à exécuter des actions. Nous vous recommandons d'utiliser des ARN pour spécifier vos ressources individuelles dans la politique pour les actions qui prennent en charge les autorisations au niveau des ressources. Vous pouvez remplacer les ARN de ressource d'espace réservé par des ARN de ressources valides pour votre cas d'utilisation.

   Si une action ne prend pas en charge les autorisations au niveau des ressources, utilisez un caractère générique (*) pour spécifier que toutes les ressources peuvent être concernées par l'action. Pour savoir quels AWS services prennent en charge les autorisations au niveau des ressources, consultez la section AWS Services compatibles avec IAM. Pour obtenir la liste des actions dans chaque service et pour savoir quelles actions prennent en charge les autorisations au niveau des ressources, veuillez consulter Actions, ressources et clés de condition pour les services AWS.

2. (Facultatif) Ajoutez, modifiez ou supprimez des instructions de politique JSON dans le modèle. Pour en savoir plus sur l'écriture de politiques JSON, veuillez consulter Creating IAM policies (Création de politiques IAM) (console).
3. Lorsque vous avez terminé de personnaliser le modèle de politique, vous disposez des options suivantes :

   - (Facultatif) Vous pouvez copier le JSON dans le modèle pour l'utiliser séparément en dehors de la page Generated policy (Politique générée). Par exemple, si vous souhaitez utiliser le JSON pour créer une politique dans un autre compte. Si la politique de votre modèle dépasse la limite de 6 144 caractères pour les politiques JSON, la politique est divisée en plusieurs politiques.
   - Cliquez sur Next (Suivant) pour consulter et créer une politique gérée dans le même compte.

Étape 4 : Examiner et créer une politique gérée

Si vous disposez des autorisations nécessaires pour créer et attacher des politiques IAM, vous pouvez créer une politique gérée à partir de la politique qui a été générée. Vous pouvez ensuite attacher la politique à un utilisateur ou à un rôle de votre compte.

Pour examiner et créer une politique

1. Dans la page Review and create managed policy (Examiner et créer une politique), spécifiez un nom sous Name et une Description (facultatif) pour la politique que vous êtes en train de créer.
2. (Facultatif) Dans la section Summary (Résumé), vous pouvez consulter les autorisations qui seront incluses dans la politique.
3. (Facultatif) Ajoutez des métadonnées à la politique en associant les balises sous forme de paires clé-valeur. Pour de plus amples informations sur l'utilisation de balises dans IAM, veuillez consulter Balisage des ressources IAM.
4. Lorsque vous avez terminé, effectuez l'une des opérations suivantes :

   - Vous pouvez attacher la nouvelle politique directement au rôle qui a été utilisé pour générer la politique. Pour ce faire, au bas de la page, cochez la case à côté de l'option Attacher la politique au ***YourRolenom***. Ensuite, sélectionnez Create and attach policy (Créer et attacher une politique).
   - Sinon, cliquez sur Create policy (Créer une politique). Vous trouverez la politique que vous avez créée dans la liste des politiques du panneau de navigation Policies (Politiques) de la console IAM.

5. Vous pouvez attacher la politique que vous avez créée à une entité de votre compte. Après avoir attaché la politique, vous pouvez supprimer toute autre politique trop étendue qui pourrait être attachée à l'entité. Pour savoir comment attacher une politique gérée, veuillez consulter Adding IAM identity permissions (Ajout d'autorisations d'identité IAM) (console).

Générer une politique en utilisant AWS CloudTrail les données d'un autre compte

Vous pouvez créer des CloudTrail pistes qui stockent les données dans des comptes centraux afin de rationaliser les activités de gouvernance. Par exemple, vous pouvez l'utiliser AWS Organizations pour créer un journal qui enregistre tous les événements pour tous les membres Comptes AWS de cette organisation. La piste appartient à un compte central. Si vous souhaitez générer une politique pour un utilisateur ou un rôle dans un compte différent de celui dans lequel sont stockées vos données de CloudTrail journal, vous devez accorder un accès entre comptes. Pour ce faire, vous avez besoin à la fois d'un rôle et d'une politique de compartiment qui accordent à IAM Access Analyzer des autorisations sur vos CloudTrail journaux. Pour plus d'informations sur la création de pistes Organisations, veuillez consulter Creating a trail for an organization (Création d'une piste pour une organisation).

Dans cet exemple, supposons que vous souhaitiez générer une politique pour un utilisateur ou un rôle dans le compte A. L' CloudTrail historique du compte A stocke les CloudTrail journaux dans un compartiment du compte B. Avant de pouvoir générer une politique, vous devez effectuer les mises à jour suivantes :

1. Choisissez un rôle existant ou créez un nouveau rôle de service qui accorde à IAM Access Analyzer l'accès au bucket du compte B (où sont stockés vos CloudTrail journaux).
2. Vérifiez votre politique de propriété d'objets de compartiment Amazon S3 et d'autorisations de compartiment dans le compte B afin que l'analyseur d'accès IAM puisse accéder aux objets du compartiment.

Étape 1 : choisir ou créer un rôle pour l'accès intercompte

- Sur l'écran Generate policy (Générer une politique), l'optionUse an existing role (Utiliser un rôle existant) est présélectionnée pour vous si un rôle avec les autorisations requises existe dans votre compte. Sinon, sélectionnez Create and use a new service role (Créer et utiliser un nouveau rôle de service). Le nouveau rôle est utilisé pour accorder à IAM Access Analyzer l'accès à votre compte de CloudTrail connexion B.

Étape 2 : vérifier la configuration du compartiment Amazon S3 dans le compte B

1. Connectez-vous à la console Amazon S3 AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/s3/.
2. Dans la liste des compartiments, choisissez le nom du compartiment dans lequel sont stockés vos CloudTrail journaux de suivi.
3. Choisissez l'onglet Permissions (Autorisations) et accédez à la section Object ownership (Propriété des objets).

   Utilisez les paramètres de propriété des objets du compartiment Amazon S3 pour contrôler la propriété des objets que vous chargez dans vos compartiments. Par défaut, lorsque d'autres Comptes AWS chargent des objets dans votre compartiment, le compte de téléchargement est propriétaire des objets. Pour générer une politique, le propriétaire du compartiment doit posséder tous les objets du compartiment. Selon votre cas d'utilisation ACL, vous pouvez modifier le réglage Object Ownership (Propriété de l'objet) pour votre compartiment. Définissez Object Ownership (Propriété de l'objet) sur l'une des options suivantes.

   - Bucket owner enforced (Propriétaire du compartiment imposé) (recommandé)
   - Bucket owner preferred (Propriétaire du compartiment préféré)

   > **Important**
   >
   > Pour générer une politique avec succès, les objets du compartiment doivent appartenir au propriétaire du compartiment. Si vous choisissez d'utiliser Bucket owner preferred (Propriétaire du compartiment préféré), vous ne pouvez générer une politique que pour la période qui suit la modification de propriété de l'objet.

   Pour en savoir plus sur la propriété des objets dans Amazon S3, veuillez consulter la rubrique Contrôle de la propriété des objets et désactivation des ACL pour votre compartiment dans le Guide de l'utilisateur d'Simple Storage Service (Amazon S3).
4. Ajoutez des autorisations à votre politique de compartiment Amazon S3 dans le compte B afin d'autoriser l'accès au rôle dans le compte A.

   L'exemple de politique suivant autorise `ListBucket` et `GetObject` pour le compartiment nommé `DOC-EXAMPLE-BUCKET`. Il autorise l'accès si le rôle accédant au compartiment appartient à un compte de votre organisation et a un nom commençant par

`AccessAnalyzerMonitorServiceRole`. L'utilisation de l'`Resource`élément `Condition` en `aws:PrincipalArn`tant qu'élément garantit que le rôle ne peut accéder à l'activité du compte que s'il appartient au compte A. Vous pouvez le `DOC-EXAMPLE-BUCKET` remplacer par le nom de votre bucket, `optional-prefix` par un préfixe facultatif pour le bucket et `organization-id` par l'ID de votre organisation.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PolicyGenerationBucketPolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/optional-prefix/AWSLogs/organization-id/
${aws:PrincipalAccount}/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "organization-id"
        },
        "StringLike": {
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
    }
  ]
}
```

5. Si vous chiffrez vos journaux en utilisant AWS KMS, mettez à jour votre politique de AWS KMS clé dans le compte sur lequel vous stockez les CloudTrail journaux afin d'autoriser IAM Access Analyzer à utiliser votre clé, comme indiqué dans l'exemple de politique suivant. Remplacez `CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN` par l'ARN pour votre piste et `organization-id` par l'ID de votre organisation.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "kms:Decrypt",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:cloudtrail:arn":
 "CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN",
          "aws:PrincipalOrgID": "organization-id"
        },
        "StringLike": {
          "kms:ViaService": [
            "access-analyzer.*.amazonaws.com",
            "s3.*.amazonaws.com"
          ]
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
    }
  ]
}
```

Génération d'une politique basée sur CloudTrail l'activité (AWS CLI)

Vous pouvez utiliser les commandes suivantes pour générer une politique à l'aide de la AWS CLI.

Pour générer une politique

- analyseur d'accès AWS start-policy-generation

Pour afficher une politique générée

- analyseur d'accès AWS get-generated-policy

Pour annuler une demande de génération de politique

- analyseur d'accès AWS cancel-policy-generation

Pour afficher la liste des demandes de génération de politique

- analyseur d'accès AWS list-policy-generations

Générer une politique basée sur CloudTrail l'activité (AWS API)

Vous pouvez utiliser les opérations suivantes pour générer une politique à l'aide de l' AWS API.

Pour générer une politique

- StartPolicyGénération

Pour afficher une politique générée

- GetGeneratedPolitique

Pour annuler une demande de génération de politique

- CancelPolicyGénération

Pour afficher la liste des demandes de génération de politique

- ListPolicyGénérations

Services de génération d'une politique d'Analyseur d'accès IAM

Le tableau suivant répertorie les AWS services pour lesquels IAM Access Analyzer génère des politiques avec des informations au niveau de l'action. Pour obtenir la liste des actions de chaque service, consultez la section Actions, ressources et clés de condition pour les AWS services dans la référence d'autorisation de service.

Service	Préfixe de service
AWS Identity and Access Management Access Analyzer	access-analyzer
AWS Account Management	compte
AWS Certificate Manager	acm
Amazon Managed Workflows for Apache Airflow	airflow
AWS Amplify	amplify
AWS Amplify Générateur d'interface utilisateur	amplifyuibuilder
Amazon AppIntegrations	app-integrations
AWS AppConfig	appconfig
Amazon AppFlow	appflow
AWS Profileur des coûts d'application	application-cost-profiler
Informations sur les CloudWatch applications Amazon	applicationinsights
AWS App Mesh	appmesh
Amazon AppStream 2.0	appstream
AWS AppSync	appsync
Amazon Managed Service for Prometheus	aps
Amazon Athena	athena
AWS Audit Manager	auditmanager
AWS Auto Scaling	scalabilité automatique
AWS Marketplace	aws-marketplace

Service	Préfixe de service
AWS Backup	sauvegarde
AWS Batch	lot
Amazon Braket	braket
AWS Budgets	Budgets
AWS Cloud9	Cloud9
AWS CloudFormation	cloudformation
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm
Amazon CloudSearch	cloudsearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch
AWS CodeArtifact	codeartifact
AWS CodeDeploy	codedeploy
Amazon CodeGuru Profiler	codeguru-profiler
CodeGuru Réviseur Amazon	codeguru-reviewer
AWS CodePipeline	codepipeline
AWS CodeStar	codestar
AWS CodeStar Notifications	codestar-notifications
Amazon Cognito Identity	cognito-identity
Groupes d'utilisateurs Amazon Cognito	cognito-idp

Service	Préfixe de service
Amazon Cognito Sync	cognito-sync
Amazon Comprehend Medical	comprehendmedical
AWS Compute Optimizer	compute-optimizer
AWS Config	config
Amazon Connect	connect
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	dataexchange
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax
AWS Device Farm	devicefarm
Amazon DevOps Guru	devops-guru
AWS Direct Connect	directconnect
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms
Clusters élastiques Amazon DocumentDB	docdb-elastic
AWS Directory Service	ds
Amazon DynamoDB	dynamodb
Amazon Elastic Block Store	ebs
Amazon Elastic Compute Cloud	ec2

Service	Préfixe de service
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Public	ecr-public
Amazon Elastic Container Service	ecs
Amazon Elastic Kubernetes Service	eks
Amazon Elastic Inference	elastic-inference
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	elasticbeanstalk
Amazon Elastic File System	elasticfilesystem
Elastic Load Balancing	elasticloadbalancing
Amazon Elastic Transcoder	elastictranscoder
Amazon EMR sur EKS (conteneurs EMR)	emr-containers
Amazon EMR sans serveur	emr-serverless
Amazon OpenSearch Service	es
Amazon EventBridge	événements
Amazon, CloudWatch évidemment	evidently
Amazon FinSpace	finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	frauddetector

Service	Préfixe de service
Amazon FSx	fsx
Amazon GameLift	gamelift
Amazon Location Service	geo
Amazon S3 Glacier	glacier
Amazon Managed Grafana	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	groundstation
Amazon GuardDuty	guardduty
AWS HealthLake	healthlake
Amazon Honeycode	honeycode
AWS Identity and Access Management	iam
AWS Boutique d'identités	identitystore
EC2 Image Builder	imagebuilder
Amazon Inspector Classic	inspector
Amazon Inspector	inspector2
AWS IoT	iot
AWS IoT Analytics	iotanalytics
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub

Service	Préfixe de service
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	iottwinmaker
AWS IoT Wireless	iotwireless
Amazon Interactive Video Service	ivs
Service de vidéo interactive Amazon Chat	ivschat
Streaming géré par Amazon pour Apache Kafka	kafka
Amazon Managed Streaming for Kafka Connect	kafkaconnect
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalytics
AWS Key Management Service	kms
AWS Lambda	lambda
Amazon Lex	lex
AWS License Manager Gestionnaire des abonnements Linux	license-manager-linux-subscriptions
Amazon Lightsail	lightsail
Amazon CloudWatch Logs	journaux
Amazon Lookout for Equipment	lookoutequipment
Amazon Lookout for Metrics	lookoutmetrics
Amazon Lookout for Vision	lookoutvision
AWS Mainframe Modernization	m2

Service	Préfixe de service
Amazon Managed Blockchain	managedblockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaStore	mediastore
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB for Redis	memorydb
AWS Application Migration Service	mgn
AWS Migration Hub	mgh
AWS Migration Hub Strategy Recommendations	migrationhub-strategy
Amazon Pinpoint	mobiletargeting
Amazon MQ	mq
AWS Network Manager	networkmanager
Amazon Nimble Studio	nimble
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	outposts
AWS Organizations	organisations

Service	Préfixe de service
AWS Panorama	panorama
AWS Performance Insights	pi
Amazon EventBridge Pipes	pipes
Amazon Polly	polly
Amazon Connect Customer Profiles	profile
Amazon QLDB	qldb
AWS Resource Access Manager	ram
AWS Corbeille	rbin
Amazon Relational Database Service	rds
Amazon Redshift	redshift
API de données Amazon Redshift	redshift-data
AWS Migration Hub Refactor Spaces	refactor-spaces
Amazon Rekognition	rekognition
AWS Resilience Hub	resiliencehub
Explorateur de ressources AWS	resource-explorer-2
AWS Resource Groups	resource-groups
AWS RoboMaker	robomaker
AWS Identity and Access Management Des rôles n'importe où	rolesanywhere
Amazon Route 53	route53

Service	Préfixe de service
Amazon Route 53 Recovery Controls	route53-recovery-control-config
Amazon Route 53 Recovery Readiness	route53-recovery-readiness
Amazon Route 53 Resolver	route53resolver
AWS CloudWatch RHUM	rum
Amazon Simple Storage Service	s3
Amazon S3 on Outposts	s3-outposts
Fonctionnalités SageMaker géospatiales d'Amazon	sagemaker-geospatial
Savings Plans	savingsplans
EventBridge Schémas Amazon	schemas
Amazon SimpleDB	sdb
AWS Secrets Manager	secretsmanager
AWS Security Hub	securityhub
Amazon Security Lake	securitylake
AWS Serverless Application Repository	serverlessrepo
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Amazon Simple Email Service	ses
AWS Shield	shield

Service	Préfixe de service
AWS Signer	signer
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Service de messages SMS et vocaux Amazon Pinpoint	sms-voice
AWS Snowball	snowball
Amazon Simple Queue Service	sqs
AWS Systems Manager	ssm
AWS Systems Manager Incident Manager	ssm-incidents
Gestionnaire de systèmes AWS pour SAP	ssm-sap
AWS Step Functions	states
AWS Security Token Service	sts
Amazon Simple Workflow Service	swf
Amazon CloudWatch Synthetics	synthetics
AWS Resource Groups Tagging API	balise
Amazon Textract	textract
Amazon Timestream	timestream
AWS Générateur de réseaux de télécommunications	tnb
Amazon Transcribe	transcribe
AWS Transfer Family	transfert
Amazon Translate	translate

Service	Préfixe de service
Amazon Connect Voice ID	voiceid
Amazon VPC Lattice	vpc-lattice
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Amazon Connect Wisdom	wisdom
Amazon WorkLink	worklink
Amazon WorkSpaces	espaces de travail
AWS X-Ray	xray

Quotas de l'IAM Access Analyzer

IAM Access Analyzer a les quotas suivants :

Ressource	Quota par défaut	Quota maximal
Nombre maximum d'analyseurs au niveau du compte par type d'analyseur et par région Compte AWS	1	1
Nombre maximum d'analyseurs au niveau de l'organisation par type d'analyseur et par région Compte AWS	5	20[1]
Règles d'archivage maximum par analyseur	100 Chaque règle d'archivage peut comporter jusqu'à 20 valeurs par critère.	1 000[1]

Ressource	Quota par défaut	Quota maximal
Nombre maximum de prévisualisations d'accès par analyseur et par heure	1 000	1 000
AWS CloudTrail fichiers journaux traités selon les générations de politiques	100 000	100 000
Générations de politique simultanées	1	1
Taille des AWS CloudTrai l données de génération de politiques	25 Go	25 Go
Échelle de AWS CloudTrai l temps de génération des politiques	90 jours	90 jours
Générations de politiques par jour	Afrique (Le Cap) : 5 Asie-Pacifique (Hong Kong) : 5 Europe (Milan) : 5 Moyen-Orient (Bahreïn) : 5 Toutes les autres régions prises en charge : 50 **Note** Les demandes de génération de politique annulées s'appliquent au quota quotidien.	Afrique (Le Cap) : 5 Asie-Pacifique (Hong Kong) : 5 Europe (Milan) : 5 Moyen-Orient (Bahreïn) : 5 Toutes les autres régions prises en charge : 50

[1]Certains quotas sont configurables par le client via l’option Service Quotas (service des quotas).

Dépannage IAM

Si vous rencontrez des problèmes liés à un refus d'accès ou des problèmes similaires lors de l'utilisation d' AWS Identity and Access Management (IAM), reportez-vous aux rubriques de cette section.

Rubriques

- Résolution des problèmes généraux d'IAM
- Résolution des problèmes liés aux messages d'erreur d'accès rejeté
- Résolution des problèmes liés aux politiques IAM
- Résolution des problèmes liés aux clés de sécurité FIDO
- Résolution des problèmes liés aux rôles IAM
- Dépannage d'IAM et Amazon EC2
- Résolution des problèmes liés à IAM et Amazon S3
- Résolution des problèmes liés à la fédération SAML 2.0 avec AWS

Résolution des problèmes généraux d'IAM

Utilisez les informations ici pour vous aider à diagnostiquer et résoudre les problèmes courants lorsque vous utilisez AWS Identity and Access Management (IAM).

Problèmes

- Je ne peux pas me connecter à mon compte AWS
- J'ai perdu mes clés d'accès
- Les variables de la politique ne fonctionnent pas
- Les modifications que j'apporte ne sont pas toujours visibles immédiatement
- Je ne suis pas autorisé à effectuer : iam : MFAdevice DeleteVirtual
- Comment créer des utilisateurs IAM en toute sécurité ?
- Ressources supplémentaires

Je ne peux pas me connecter à mon compte AWS

Vérifiez que vous disposez des informations d'identification correctes et que vous utilisez la bonne méthode pour vous connecter. Pour plus d'informations, consultez Résolution des problèmes de connexion dans le Guide de l'utilisateur Connexion à AWS .

J'ai perdu mes clés d'accès

Les clés d'accès se composent de deux parties :

- Identificateur de clé d'accès. Cet élément n'est pas secret et peut être affiché dans la console IAM partout où les clés d'accès sont répertoriées, comme sur la page de récapitulatif de l'utilisateur.
- Clé d'accès secrète. Cette partie est fournie lorsque vous créez initialement la paire de clés d'accès. Tout comme un mot de passe, elle ne peut pas être récupérée ultérieurement. Si vous perdez votre clé d'accès secrète, vous devez créer une nouvelle paire de clés d'accès. Si vous disposez déjà du nombre maximum de clés d'accès, vous devez supprimer une paire existante avant de pouvoir en créer une autre.

Pour plus d’informations, consultez Réinitialisation de mots de passe ou de clés d'accès perdus ou oubliés pour AWS.

Les variables de la politique ne fonctionnent pas

- Vérifiez que toutes les politiques incluant des variables contiennent le numéro de version suivant : `"Version": "2012-10-17"`. Sans le numéro de version approprié, les variables ne sont pas remplacées lors de l'évaluation. Au contraire, les variables sont évaluées littéralement. Toutes les politiques n'incluant pas de variables continueront de fonctionner si vous incluez le numéro de version le plus récent.

  Un élément de politique `Version` est différent d'une version de politique. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. En revanche, une version de politique est créée lorsque vous apportez des modifications à une politique gérée par le client dans IAM. La politique modifiée ne remplace pas la politique existante. À la place, IAM crée une nouvelle version de la politique gérée. Pour en savoir plus sur l'élément de politique `Version`, consultez Éléments de politique JSON IAM : Version. Pour en savoir plus sur les versions de politiques, consultez the section called “Gestion des versions des politiques IAM”.

- Vérifiez que vos variables de politique sont dans la casse correcte. Pour plus d'informations, veuillez consulter Éléments des politiques IAM : variables et balises.

Les modifications que j'apporte ne sont pas toujours visibles immédiatement

En tant que service auquel on accède avec des ordinateurs situés dans des centres de données du monde entier, IAM utilise un modèle d'informatique distribuée appelé cohérence éventuelle. Toute modification apportée à IAM (ou à d'autres AWS services), y compris les balises utilisées dans le contrôle d'accès basé sur les attributs (ABAC), met du temps à être visible depuis tous les points de terminaison possibles. Une partie du retard s'explique par le temps requis pour envoyer les données d'un serveur à un autre, d'une zone de réplication à une autre et d'une région à une autre dans le monde entier. IAM utilise également la mise en cache pour améliorer les performances mais, dans certains cas, cela peut ralentir le processus. La modification peut ne pas être visible tant que les données mises en cache précédemment n'arrivent pas à expiration.

Vous devez concevoir vos applications globales de sorte qu'elles tiennent compte de ces retards potentiels. Assurez-vous qu'elles fonctionnent comme prévu, même lorsqu'une modification effectuée à un emplacement n'est pas visible instantanément à un autre. Les modifications peuvent être la création ou la mise à jour d'utilisateurs, de groupes, de rôles ou de politiques. Nous vous recommandons de ne pas inclure ce type de modifications IAM dans les chemins de code critique et haute disponibilité de votre application. Au lieu de cela, procédez aux modifications IAM dans une routine d'initialisation ou d'installation distincte que vous exécutez moins souvent. Veillez également à vérifier que les modifications ont été propagées avant que les processus de production en dépendent.

Pour plus d'informations sur la manière dont d'autres AWS services sont affectés par cette situation, consultez les ressources suivantes :

- Amazon DynamoDB :Quel est le modèle de cohérence d'Amazon DynamoDB ? dans la FAQ DynamoDB, et Cohérence en lecture dans le Manuel du développeur Amazon DynamoDB.
- Amazon EC2 : Cohérence éventuelle EC2 dans la Référence d'API Amazon EC2
- Amazon EMR : garantir la cohérence lors de l'utilisation d'Amazon S3 et d'Amazon Elastic MapReduce pour les flux de travail ETL dans le cadre du blog AWS Big Data
- Amazon Redshift :gestion de la cohérence des données dans le guide du développeur de la base de données Amazon Redshift
- Amazon S3 : modèle de cohérence de données Amazon S3 dans le guide d'utilisateur du service de stockage simple d'Amazon

Je ne suis pas autorisé à effectuer : iam : MFAdevice DeleteVirtual

L'erreur suivante peut s'afficher lorsque vous tentez d'attribuer ou de supprimer un dispositif MFA virtuel pour vous-même ou d'autres personnes :

```
User: arn:aws:iam::123456789012:user/Diego is not authorized to perform:
 iam:DeleteVirtualMFADevice on resource: arn:aws:iam::123456789012:mfa/Diego with an
 explicit deny
```

Elle peut se produire si quelqu'un a commencé à assigner un dispositif MFA virtuel à un utilisateur dans la console IAM et a annulé le processus. Cela crée un dispositif MFA virtuel pour l'utilisateur dans IAM, mais ne l'affecte jamais à l'utilisateur. Vous devez supprimer le dispositif MFA virtuel existant avant de pouvoir en créer un nouveau portant le même nom.

Pour résoudre ce problème, l'administrateur ne doit pas modifier les autorisations de politique. L'administrateur doit plutôt utiliser l' AWS API AWS CLI or pour supprimer le dispositif MFA virtuel existant mais non attribué.

Supprimer un dispositif MFA virtuel existant, mais non attribué

1. Affichez les dispositifs MFA virtuels de votre compte.
   - AWS CLI: `aws iam list-virtual-mfa-devices`
   - AWS API : `ListVirtualMFADevices`
2. Dans la réponse, localisez l'ARN du dispositif MFA virtuel pour l'utilisateur que vous essayez de réparer.
3. Supprimez le dispositif MFA virtuel.
   - AWS CLI: `aws iam delete-virtual-mfa-device`
   - AWS API : `DeleteVirtualMFADevice`

Comment créer des utilisateurs IAM en toute sécurité ?

Si certains de vos employés ont besoin d'accéder à AWS, vous pouvez choisir de créer des utilisateurs IAM ou d'utiliser IAM Identity Center pour l'authentification. Si vous utilisez IAM, il est AWS recommandé de créer un utilisateur IAM et de communiquer en toute sécurité les informations d'identification à l'employé. Si vous ne vous trouvez pas physiquement à côté de votre employé, utilisez un flux de travail sécurisé pour communiquer les informations d'identification à celui-ci.

Utilisez le flux de travail suivant pour créer en toute sécurité un utilisateur dans IAM :

1. Créez un utilisateur à l'aide de la AWS Management Console. Choisissez d'accorder l' AWS Management Console accès à l'aide d'un mot de passe généré automatiquement. Si nécessaire, cochez la case Les utilisateurs doivent créer un mot de passe à leur prochaine connexion. N'ajoutez pas de politique d'autorisations à l'utilisateur tant que celui-ci n'a pas modifié son mot de passe.
2. Une fois l'utilisateur ajouté, copiez l'URL de connexion, le nom d'utilisateur et le mot de passe du nouvel utilisateur. Pour afficher le mot de passe, choisissez afficher.
3. Envoyez le mot de passe à votre employé à l'aide d'une méthode de communication sécurisée dans votre entreprise, par exemple, par e-mail, chat ou système de tickets. Séparément, fournissez à vos utilisateurs le lien de la console utilisateur IAM et leur nom d'utilisateur. Demandez à l'employé de vous confirmer qu'il peut se connecter correctement avant de lui accorder des autorisations.
4. Une fois que l'employé vous a confirmé cela, ajoutez les autorisations dont il a besoin. Pour des raisons de sécurité, il est recommandé d'ajouter une politique qui exige que l'utilisateur s'authentifie à l'aide de MFA pour gérer ses informations d'identification. Pour un exemple de politique, veuillez consulter AWS: permet aux utilisateurs IAM authentifiés par MFA de gérer leurs propres informations d'identification sur la page Informations d'identification de sécurité.

Ressources supplémentaires

Les ressources suivantes peuvent vous aider à résoudre les problèmes au fur et à mesure que vous travaillez avec AWS.

- AWS CloudTrail Guide de l'utilisateur : AWS CloudTrail permet de suivre l'historique des appels d'API effectués AWS et de stocker ces informations dans des fichiers journaux. Vous pouvez ainsi identifier les utilisateurs et les comptes ayant accédé aux ressources de votre compte lors des appels, déterminer les actions demandées, etc. Pour plus d’informations, consultez Journalisation des appels IAM et AWS STS API avec AWS CloudTrail.
- AWS Centre de connaissances — Trouvez des FAQ et des liens vers d'autres ressources pour vous aider à résoudre les problèmes.
- AWS Centre de support — Bénéficiez d'une assistance technique.
- AWS Centre de support Premium — Bénéficiez d'un support technique haut de gamme.

Résolution des problèmes liés aux messages d'erreur d'accès rejeté

Les erreurs d'accès refusé apparaissent en cas de refus AWS explicite ou implicite d'une demande d'autorisation. Un refus explicite se produit lorsqu'une politique contient une `Deny` déclaration pour une AWS action spécifique. Un refus implicite se produit lorsqu'il n'y a ni d'instruction `Deny` applicable ni d'instruction `Allow` applicable. Comme une politique IAM refuse un principal IAM par défaut, elle doit autoriser explicitement le principal à effectuer une action. Dans le cas contraire, la politique refuse implicitement l'accès. Pour plus d'informations, consultez Différence entre les refus explicites et implicites.

Si plusieurs politiques du même type de politique refusent une demande d'autorisation, AWS ne précise pas le nombre dans le message d'erreur d'accès refusé. Si plusieurs types de politiques refusent une demande d'autorisation, n' AWS inclut qu'un seul de ces types de politique dans le message d'erreur.

> **Important**
>
> Vous rencontrez des difficultés pour vous connecter à AWS ? Assurez-vous que vous êtes sur la bonne page de connexion AWS pour votre type d'utilisateur. Si vous êtes le Utilisateur racine d'un compte AWS (propriétaire du compte), vous pouvez vous connecter à AWS l'aide des informations d'identification que vous avez définies lors de la création du Compte AWS. Si vous êtes un utilisateur IAM, votre administrateur de compte peut vous fournir les informations d'identification que vous pouvez utiliser pour vous connecter à AWS. Si vous avez besoin d'assistance, n'utilisez pas le lien de commentaires sur cette page, car le formulaire est reçu par l'équipe de AWS documentation AWS Support. Sur la page Contactez-nous, sélectionnez Impossible de vous connecter à votre AWS compte, puis choisissez l'une des options d'assistance disponibles.

Je reçois un « accès refusé » lorsque je fais une demande à un AWS service

- Vérifiez si le message d'erreur inclut le type de politique responsable du refus d'accès. Par exemple, si l'erreur indique que l'accès est refusé en raison d'une politique de contrôle des services (SCP), vous pouvez vous concentrer sur le dépannage des problèmes SCP. Une fois que vous connaissez le type de politique, vous pouvez rechercher une instruction de refus ou

une autorisation manquante sur l'action spécifique dans les politiques de ce type. Si le message d'erreur n'indique pas le type de politique responsable du refus d'accès, utilisez les autres instructions de cette section pour effectuer un dépannage plus approfondi.

- Vérifiez que vous avez l'autorisation de politique basée sur l'identité d'appeler l'action et la ressource que vous avez demandées. Si des conditions sont définies, vous devez également respecter ces conditions lorsque vous envoyez la demande. Pour plus d'informations sur l'affichage ou la modification de politiques pour un utilisateur, un groupe ou un rôle IAM, veuillez consulter Gestion des politiques IAM.
- S'il AWS Management Console renvoie un message indiquant que vous n'êtes pas autorisé à effectuer une action, vous devez contacter votre administrateur pour obtenir de l'aide. Votre administrateur vous a fourni vos informations de connexion ou votre lien de connexion.

  L'exemple d'erreur suivant se produit quand l'utilisateur IAM `mateojackson` tente d'utiliser la console pour afficher des informations détaillées sur une ressource *`my-example-widget`* fictive, mais ne dispose pas des autorisations `widgets:`*`GetWidget`* fictives.

  ```
  User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
   widgets:GetWidget on resource: my-example-widget
  ```

  Dans ce cas, Mateo doit demander à son administrateur de mettre à jour ses politiques pour lui permettre d'accéder à la ressource *`my-example-widget`* à l'aide de l'action `widgets:`*`GetWidget`*.
- Essayez-vous d'accéder à un service prenant en charge des politiques basées sur les ressources, tel que Amazon S3, Amazon SNS ou Amazon SQS ? Si tel est le cas, vérifiez que la politique vous spécifie comme principal et vous accorde l'accès. Si vous effectuez une demande à un service dans votre compte, vos politiques basées sur une identité ou sur les ressources peuvent vous accorder l'autorisation. Si vous effectuez une demande à un service dans un autre compte, vos politiques basées sur une identité et sur les ressources doivent vous accorder l'autorisation. Pour afficher les services qui prennent en charge les politiques basées sur les ressources, veuillez consulter AWS services qui fonctionnent avec IAM.
- Si votre politique inclut une condition avec une paire clé-valeur, vérifiez-la attentivement. Les exemples incluent la clé de condition `aws:RequestTag/`*`tag-key`*globale AWS KMS `kms:EncryptionContext:`*`encryption_context_key`*, la clé de condition et la clé de `ResourceTag/`*`tag-key`* condition prises en charge par plusieurs services. Assurez-vous que le nom de clé ne correspond pas à plusieurs résultats. Les noms de clé de condition n'étant pas sensibles à la casse, une condition qui vérifie une clé nommée `foo` correspond à `foo`, `Foo`, ou

F00. Si votre demande comprend plusieurs paires clé-valeur avec des noms de clé qui ne diffèrent que par la casse, votre accès pourrait être inopinément refusé. Pour plus d'informations, veuillez consulter Éléments de politique JSON IAM : Condition.

- Si vous disposez d'une limite d'autorisations, vérifiez que la politique utilisée pour cette dernière autorise votre demande. Si vos politiques basées sur l'identité permettent la demande, mais pas votre limite d'autorisations, la demande est refusée. Une limite d'autorisations contrôle les autorisations maximum dont peut disposer un principal IAM (utilisateur ou rôle). Les politiques basées sur les ressources ne sont pas limitées par des limites d'autorisations. Les limites d'autorisations ne sont pas courantes. Pour plus d'informations sur le mode AWS d'évaluation des politiques, consultezLogique d'évaluation de politiques.
- Si vous signez des demandes manuellement (sans utiliser les kits SDK AWS), vérifiez que vous avez correctement signé la demande.

Je reçois un message d'accès refusé lorsque j'effectue une demande avec des informations d'identification de sécurité temporaires

- Tout d'abord, assurez-vous que l'accès ne vous est pas refusé pour une raison non liée à vos informations d'identification temporaires. Pour plus d'informations, veuillez consulter Je reçois un « accès refusé » lorsque je fais une demande à un AWS service.
- Pour vérifier que le service accepte les informations d'identification de sécurité temporaires, veuillez consulter AWS services qui fonctionnent avec IAM.
- Vérifiez que vos demandes sont signées correctement et que la demande est correctement formée. Pour en savoir plus, veuillez consulter la documentation de votre boîte à outils ou Utilisation d'informations d'identification temporaires avec des ressources AWS.
- Vérifiez que vos informations d'identification de sécurité temporaires ne sont pas arrivées à expiration. Pour plus d'informations, veuillez consulter Informations d'identification de sécurité temporaires dans IAM.
- Vérifiez que l'utilisateur ou le rôle IAM dispose des autorisations adéquates. Les autorisations pour les informations d'identification de sécurité temporaires sont dérivées d'un utilisateur ou d'un rôle IAM. Par conséquent, les autorisations sont limitées à celles qui sont accordées au rôle dont vous avez utilisé les informations d'identification temporaires. Pour de plus amples informations sur la définition des autorisations des informations d'identification de sécurité temporaires, veuillez consulter Contrôle des autorisations affectées aux informations d'identification de sécurité temporaires.

- Si vous avez endossé un rôle, votre session de rôle peut être limitée par les politiques de session. Lorsque vous demandez des informations d'identification de sécurité temporaires par le biais d'un programme AWS STS, vous pouvez éventuellement transmettre des politiques de session en ligne ou gérées. Les politiques de session sont des politiques avancées que vous transmettez en tant que paramètre lorsque vous créez par programmation une session d'informations d'identification temporaires pour un rôle. Vous pouvez transmettre un seul document de politique de session en ligne JSON à l'aide du paramètre `Policy`. Vous pouvez utiliser le paramètre `PolicyArns` pour spécifier jusqu'à 10 politiques de session gérées. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité du rôle et des politiques de session. Sinon, si votre administrateur ou un programme personnalisé vous fournit des informations d'identification temporaires, ils peuvent avoir inclus une politique de session pour limiter votre accès.
- Si vous êtes un utilisateur fédéré, votre session peut être limitée par les politiques de session. Vous devenez un utilisateur fédéré en vous connectant en AWS tant qu'utilisateur IAM, puis en demandant un jeton de fédération. Pour de plus amples informations sur les utilisateurs fédérés, veuillez consulter GetFederationToken : fédération via un broker d'identité personnalisé. Si vous ou votre broker d'identité avez transmis des politiques de session lorsque vous avez demandé un jeton de session, votre session est limitée par ces politiques. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité de l'utilisateur IAM et des politiques de session. Pour de amples informations sur les politiques de session, veuillez consulter Politiques de session.
- Si vous accédez à une ressource disposant d'une politique basée sur une ressource à l'aide d'un rôle, vérifiez que la politique accorde des autorisations au rôle. Par exemple, la politique suivante autorise `MyRole` du compte `111122223333` à accéder à `MyBucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "S3BucketPolicy",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::111122223333:role/MyRole"]},
    "Action": ["s3:PutObject"],
    "Resource": ["arn:aws:s3:::MyBucket/*"]
  }]
}
```

Exemples de messages d'erreur d'accès refusé

La plupart des messages d'erreur d'accès refusé se présentent sous le format `User` *`user`* `is not authorized to perform` *`action`* `on` *`resource`* `because` *`context`*. Dans cet exemple, *`user`* (utilisateur) est le Amazon Resource Name (ARN) qui ne reçoit pas d'accès, *`action`* est l'action de service refusée par la politique, et *`ressource`* est l'ARN de la ressource sur laquelle la politique agit. Le champ de *`context`* (contexte) représente un contexte supplémentaire sur le type de politique qui explique pourquoi l'accès est refusé.

Lorsqu'une politique refuse explicitement l'accès parce qu'elle contient une `Deny` instruction, elle AWS inclut la phrase `with an explicit deny in a` *`type`* `policy` dans le message d'erreur d'accès refusé. Lorsque la politique refuse implicitement l'accès, AWS inclut la phrase `because no` *`type`* `policy allows the` *`action`* `action` dans le message d'erreur d'accès refusé.

> **Note**
>
> Certains AWS services ne prennent pas en charge ce format de message d'erreur de refus d'accès. Le contenu des messages d'erreur d'accès refusé peut varier en fonction du service à l'origine de la demande d'autorisation.

Les exemples suivants montrent le format pour différents types de messages d'erreur d'accès refusé.

Accès refusé en raison d'une politique de contrôle des services - refus implicite

1. Vérifiez la présence d'une instruction `Allow` manquante pour l'action dans vos politiques de contrôle des services (SCP). Pour l'exemple suivant, l'action est `codecommit:ListRepositories`.
2. Mettez à jour votre SCP en ajoutant l'instruction `Allow`. Pour plus d'information, consultez la section relative aux mises à jour Over-the-Air dans le AWS Organizations User Guide.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no service control policy allows the
 codecommit:ListRespositories action
```

Accès refusé en raison d'une politique de contrôle des services - refus explicite

1. Vérifiez la présence d'une instruction `Deny` pour l'action dans vos politiques de contrôle des services (SCP). Pour l'exemple suivant, l'action est `codecommit:ListRepositories`.
2. Mettez à jour votre SCP en supprimant l'instruction `Deny`. Pour plus d'information, consultez la section relative aux mises à jour Over-the-Air dans le AWS Organizations User Guide.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories with an explicit deny in a service control policy
```

Accès refusé en raison d'une politique de point de terminaison d'un VPC : refus implicite

1. Vérifiez l'absence d'une instruction `Allow` pour l'action dans vos politiques de point de terminaison de cloud privé virtuel (VPC). Pour l'exemple suivant, l'action est `codecommit:ListRepositories`.
2. Mettez à jour la politique de point de terminaison d'un VPC en ajoutant l'instruction `Allow`. Pour plus d'informations, veuillez consulter la rubrique Mise à jour d'une politique de point de terminaison d'un VPC dans le Guide AWS PrivateLink .

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no VPC endpoint policy allows the
 codecommit:ListRepositories action
```

Accès refusé en raison d'une politique de point de terminaison d'un VPC : refus explicite

1. Vérifiez la présence d'une instruction `Deny` explicite pour l'action dans vos politiques de point de terminaison de cloud privé virtuel (VPC). Pour l'exemple suivant, l'action est `codedeploy:ListDeployments`.
2. Mettez à jour la politique de point de terminaison d'un VPC en supprimant l'instruction `Deny`. Pour plus d'informations, veuillez consulter la rubrique Mise à jour d'une politique de point de terminaison d'un VPC dans le Guide AWS PrivateLink .

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
```

```
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* with an explicit deny in a VPC endpoint policy
```

Accès refusé en raison d'une limite des autorisations : refus implicite

1. Vérifiez l'absence d'une instruction `Allow` pour l'action dans votre limite des autorisations. Pour l'exemple suivant, l'action est `codedeploy:ListDeployments`.
2. Mettez à jour votre limite des autorisations en ajoutant l'instruction `Allow` à votre politique IAM. Pour plus d'informations, consultez Limites d'autorisations pour les entités IAM et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* because no permissions boundary allows the
 codedeploy:ListDeployments action
```

Accès refusé en raison d'une limite des autorisations : refus explicite

1. Vérifiez la présence d'une instruction `Deny` explicite pour l'action dans votre limite des autorisations. Pour l'exemple suivant, l'action est `sagemaker:ListModels`.
2. Mettez à jour votre limite des autorisations en supprimant l'instruction `Deny` de votre politique IAM. Pour plus d'informations, consultez Limites d'autorisations pour les entités IAM et Modification de politiques IAM.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
sagemaker:ListModels with an explicit deny in a permissions boundary
```

Accès refusé en raison de politiques de session : refus implicite

1. Vérifiez l'absence d'une instruction `Allow` pour l'action dans vos politiques de session. Pour l'exemple suivant, l'action est `codecommit:ListRepositories`.
2. Mettez à jour votre politique de session en ajoutant l'instruction `Allow`. Pour plus d'informations, veuillez consulter les rubriques Politiques de session et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
```

```
codecommit:ListRepositories because no session policy allows the
 codecommit:ListRepositories action
```

Accès refusé en raison de politiques de session : refus explicite

1. Vérifiez la présence d'une instruction `Deny` explicite pour l'action dans vos politiques de session. Pour l'exemple suivant, l'action est `codedeploy:ListDeployments`.
2. Mettez à jour votre politique de session en supprimant l'instruction `Deny`. Pour plus d'informations, veuillez consulter les rubriques Politiques de session et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* with an explicit deny in a sessions policy
```

Accès refusé en raison de politiques basées sur les ressources : refus implicite

1. Vérifiez l'absence d'une instruction `Allow` pour l'action dans votre politique basée sur les ressources. Pour l'exemple suivant, l'action est `secretsmanager:GetSecretValue`.
2. Mettez à jour votre politique en ajoutant l'instruction `Allow`. Pour plus d'informations, veuillez consulter les rubriques Politiques basées sur les ressources et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
secretsmanager:GetSecretValue because no resource-based policy allows the
 secretsmanager:GetSecretValue action
```

Accès refusé en raison de politiques basées sur les ressources : refus explicite

1. Vérifiez la présence d'une instruction `Deny` explicite pour l'action dans votre politique basée sur les ressources. Pour l'exemple suivant, l'action est `secretsmanager:GetSecretValue`.
2. Mettez à jour votre politique en supprimant l'instruction `Deny`. Pour plus d'informations, veuillez consulter les rubriques Politiques basées sur les ressources et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
secretsmanager:GetSecretValue on resource: arn:aws:secretsmanager:us-
east-1:123456789012:secret:* with an explicit deny in a resource-based policy
```

Accès refusé en raison de politiques d'approbation de rôle : refus implicite

1. Vérifiez l'absence d'une instruction `Allow` pour l'action dans votre politique d'approbation de rôle. Pour l'exemple suivant, l'action est `sts:AssumeRole`.
2. Mettez à jour votre politique en ajoutant l'instruction `Allow`. Pour plus d'informations, veuillez consulter les rubriques Politiques basées sur les ressources et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
sts:AssumeRole because no role trust policy allows the sts:AssumeRole action
```

Accès refusé en raison de politiques d'approbation de rôle : refus explicite

1. Vérifiez la présence d'une instruction `Deny` explicite pour l'action dans votre politique d'approbation de rôle. Pour l'exemple suivant, l'action est `sts:AssumeRole`.
2. Mettez à jour votre politique en supprimant l'instruction `Deny`. Pour plus d'informations, veuillez consulter les rubriques Politiques basées sur les ressources et Modification de politiques IAM.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
sts:AssumeRole with an explicit deny in the role trust policy
```

Accès refusé en raison de politiques basées sur l'identité : refus implicite

1. Vérifiez l'absence d'une instruction `Allow` pour l'action dans les politiques basées sur l'identité attachées à l'identité. Pour l'exemple suivant, l'action est `codecommit:ListRepositories` attachée à l'utilisateur `JohnDoe`.
2. Mettez à jour votre politique en ajoutant l'instruction `Allow`. Pour plus d'informations, veuillez consulter les rubriques Politiques basées sur l'identité et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no identity-based policy allows the
 codecommit:ListRepositories action
```

Accès refusé en raison de politiques basées sur l'identité : refus explicite

1. Vérifiez la présence d'une instruction `Deny` explicite pour l'action dans les politiques basées sur l'identité attachées à l'identité. Pour l'exemple suivant, l'action est `codedeploy:ListDeployments` attachée à l'utilisateur `JohnDoe`.
2. Mettez à jour votre politique en supprimant l'instruction `Deny`. Pour plus d'informations, veuillez consulter les rubriques Politiques basées sur l'identité et Modification de politiques IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* with an explicit deny in an identity-based policy
```

Accès refusé lorsqu'une requête VPC échoue en raison d'une autre politique

1. Vérifiez la présence d'une instruction `Deny` explicite pour l'action dans vos politiques de contrôle des services (SCP). Pour l'exemple suivant, l'action est `SNS:Publish`.
2. Mettez à jour votre SCP en supprimant l'instruction `Deny`. Pour plus d'information, consultez la section relative aux mises à jour Over-the-Air dans le AWS IAM Identity Center User Guide.

```
User: arn:aws:sts::111122223333:assumed-role/role-name/role-session-name is not
 authorized to perform:
SNS:Publish on resource: arn:aws:sns:us-east-1:444455556666:role-name-2
with an explicit deny in a VPC endpoint policy transitively through a service control
 policy
```

Résolution des problèmes liés aux politiques IAM

Une politique est une entité AWS qui, lorsqu'elle est attachée à une identité ou à une ressource, définit ses autorisations. AWS évalue ces politiques lorsqu'un mandant, tel qu'un utilisateur, fait une demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée. Les politiques sont stockées AWS sous forme de documents JSON attachés aux principes en tant que politiques basées sur l'identité ou aux ressources en tant que politiques basées sur les ressources. Vous pouvez attacher une politique basée sur identité à un principal (ou une identité), comme un groupe, un utilisateur ou un rôle IAM. Les politiques basées sur l'identité incluent des politiques gérées par AWS des politiques gérées par le client et des politiques en ligne. Vous pouvez créer et modifier des politiques gérées par le client à l' AWS Management Console aide des options

de l'éditeur Visual et JSON. Lorsque vous consultez une politique dans le AWS Management Console, vous pouvez voir un résumé des autorisations accordées par cette politique. Vous pouvez utiliser l'éditeur visuel et des résumés de politique pour vous aider à diagnostiquer et à corriger les erreurs courantes lors de la gestion des politiques IAM.

N'oubliez pas que toutes les politiques IAM sont stockées selon une syntaxe qui commence par les règles de notation d'JavaScript objet (JSON). Vous n'avez pas besoin de comprendre cette syntaxe pour créer ou gérer vos politiques. Vous pouvez créer et modifier une politique à l'aide de l'éditeur visuel dans l' AWS Management Console. Pour en savoir plus sur la syntaxe JSON dans les politiques IAM, veuillez consulter Syntaxe du langage de politique JSON IAM .

Rubriques de résolution des erreurs de politique IAM

- Résolution des problèmes à l'aide de l'éditeur visuel
  - Restructuration de politique
  - Choix d'un ARN de ressource dans l'éditeur visuel
  - Refus des autorisations dans l'éditeur visuel
  - Spécification de plusieurs services dans l'éditeur visuel
  - Réduction de la taille de votre politique dans l'éditeur visuel
  - Résolution des problèmes de services, d'actions ou de types de ressources non reconnus dans l'éditeur visuel
- Résolution des problèmes à l'aide des récapitulatifs de politique
  - Récapitulatif de politique manquant
  - Le récapitulatif de politique inclut des services, des actions ou des types de ressource non reconnus
  - Le service ne prend pas en charge les récapitulatifs de politique IAM
  - Ma politique n'accorde pas les autorisations escomptées
- Résolution des problèmes de gestion des politiques
  - Attacher ou détacher une politique dans un compte IAM
  - Modification des politiques pour vos identités IAM en fonction de leur activité
- Résolution des problèmes de documents de politique JSON
  - Valider vos politiques
  - Je n'ai pas d'autorisations pour la validation de politique dans l'éditeur JSON
  - Plusieurs objets de politique JSON

- Plusieurs éléments de instruction JSON
- Plusieurs éléments Effect, Action ou Resource dans un élément d'instruction JSON
- Élément de version JSON manquant

Résolution des problèmes à l'aide de l'éditeur visuel

Lorsque vous créez ou modifiez une politique gérée par le client, vous pouvez utiliser les informations de l'éditeur visuel pour vous aider à résoudre les erreurs dans votre politique. Pour voir un exemple d'utilisation de l'éditeur visuel pour la création d'un politique, veuillez consulter the section called “Contrôle de l'accès aux identités”.

Restructuration de politique

Lorsque vous créez une politique, AWS elle valide, traite et transforme la politique avant de la stocker. Lorsqu'il AWS renvoie la politique en réponse à une requête de l'utilisateur ou qu'il l'affiche dans la console, la AWS retransforme dans un format lisible par l'homme sans modifier les autorisations accordées par la politique. Cela peut entraîner des différences quant à ce que vous voyez dans l'éditeur visuel de politique ou sur l'onglet JSON : des blocs d'autorisation de l'éditeur visuel peuvent être ajoutés, supprimés ou réordonnés, et le contenu au sein d'un bloc peut être optimisé. Sous l'onglet JSON, les espaces blancs sans importance peuvent être supprimés et les éléments avec des mappes JSON peuvent être réorganisés. En outre, Compte AWS les identifiants des éléments principaux peuvent être remplacés par l'ARN du Utilisateur racine d'un compte AWS. En raison de ces changements possibles, vous ne devez pas comparer les documents de politique JSON sous forme de chaînes.

Lorsque vous créez une politique gérée par le client dans le AWS Management Console, vous pouvez choisir de travailler entièrement dans l'éditeur JSON. Si vous ne modifiez jamais l'éditeur visuel et si vous choisissez Suivant dans l'éditeur JSON, la politique a moins de chances d'être restructurée. Toutefois, si vous créez une politique et si vous utilisez l'éditeur visuel pour apporter des modifications, ou si vous choisissez Suivant dans l'option éditeur visuel, IAM peut restructurer la politique pour optimiser son affichage dans l'éditeur visuel.

Cette restructuration existe uniquement dans votre session d'édition et n'est pas enregistrée automatiquement.

Si votre politique est restructurée dans votre session d'édition, IAM détermine si la restructuration doit être enregistrée en fonction des situations suivantes :

Utilisation de cette option de l'éditeur	Si vous modifiez votre politique	Et puis choisissez Suivant dans cet onglet	Lorsque vous sélectionnez Enregistrer les modifications
Visuel	Modifié	Visuel	La stratégie est restructurée
Visuel	Modifié	JSON	La stratégie est restructurée
Visuel	Non modifié	Visuel	La stratégie est restructurée
JSON	Modifié	Visuel	La stratégie est restructurée
JSON	Modifié	JSON	La structure de la stratégie n'est pas modifiée
JSON	Non modifié	JSON	La structure de la stratégie n'est pas modifiée

IAM peut restructurer les politiques complexes ou celles qui ont des blocs d'autorisations ou des instructions qui permettent plusieurs services, types de ressources ou clés de condition.

Choix d'un ARN de ressource dans l'éditeur visuel

Lorsque vous créez ou modifiez une politique à l'aide de l'éditeur visuel, vous devez d'abord choisir un service, puis les actions de ce dernier. Si le service et les actions que vous avez sélectionnés prennent en charge la sélection de ressources spécifiques, l'éditeur visuel répertorie les types de ressources pris en charge. Vous pouvez ensuite choisir Add ARN (Ajouter un ARN) pour fournir des détails sur votre ressource. Vous pouvez choisir l'une des options suivantes afin d'ajouter un ARN pour un type de ressource.

- Utiliser le générateur d'ARN : selon le type de ressource, différents champs peuvent s'afficher pour créer votre ARN. Vous pouvez également choisir Any (Tout) pour fournir des autorisations pour

n'importe quelle valeur du paramètre spécifié. Par exemple, si vous avez sélectionné le groupe de niveau d'accès Read (Lecture) Amazon EC2, les actions de votre politique prennent en charge le type de ressource `instance`. Vous devez fournir la région, le compte et InstanceIdles valeurs de votre ressource. Si vous indiquez votre ID de compte, mais que vous sélectionnez Tout pour la région et l'ID d'instance, la politique accorde des autorisations à toutes les instances de votre compte.

- Saisir ou coller l'ARN : vous pouvez spécifier les ressources par leur Amazon Resource Name (ARN). Vous pouvez inclure un caractère générique (*) dans n'importe quel champ de l'ARN (entre chaque paire de signes deux-points). Pour plus d'informations, consultez Éléments de politique JSON IAM : Resource.

Refus des autorisations dans l'éditeur visuel

Par défaut, la politique que vous créez à l'aide de l'éditeur visuel autorise les actions que vous sélectionnez. Pour refuser les actions choisies, sélectionnez Switch to deny permissions (Basculer vers le refus des autorisations). Dans la mesure où les demandes sont refusées par défaut, nous vous recommandons, afin de garantir la sécurité, d'autoriser un utilisateur à accéder uniquement aux actions et aux ressources dont il a besoin. Vous devez créer une instruction pour refuser des autorisations uniquement si vous souhaitez remplacer une autorisation qui est également autorisée par une autre instruction ou politique. Nous vous recommandons de limiter le nombre de refus d'autorisation au minimum, car ils peuvent rendre la résolution des problèmes d'autorisation plus complexe. Pour plus d'informations sur la logique d'évaluation de politique IAM, veuillez consulter Logique d'évaluation de politiques.

> **Note**
>
> Par défaut, seul le Utilisateur racine d'un compte AWS compte a accès à toutes les ressources de ce compte. Par conséquent, si vous n'êtes pas connecté en tant qu'utilisateur racine, vous devez disposer des autorisations accordées par une politique.

Spécification de plusieurs services dans l'éditeur visuel

Lorsque vous utilisez l'éditeur visuel pour construire une politique, vous pouvez sélectionner un seul service à la fois. Il s'agit d'une bonne pratique car l'éditeur visuel vous permet ensuite de choisir parmi les actions associées à ce service. Vous pouvez ensuite choisir parmi les ressources prises

en charge par ce service et les actions sélectionnées. Cela facilite la création et la résolution des problèmes de votre politique.

Si vous connaissez la syntaxe JSON, vous pouvez également utiliser un caractère générique (*) afin de spécifier manuellement plusieurs services. Par exemple, saisissez **`Code*`** pour fournir des autorisations pour tous les services commençant par `Code`, tels que `CodeBuild` et `CodeCommit`. Toutefois, vous devez ensuite taper les ARN des actions et ressources pour finaliser votre politique. De plus, lorsque vous enregistrez votre politique, celle-ci peut être restructurée de façon à inclure chaque service dans un bloc d'autorisation distinct.

Afin d'utiliser la syntaxe JSON (comme les caractères génériques) pour les services, créez, modifiez et enregistrez votre politique à l'aide de l'option de l'éditeur JSON.

Réduction de la taille de votre politique dans l'éditeur visuel

Lorsque vous utilisez l'éditeur visuel pour créer une politique, IAM crée un document JSON pour stocker votre politique. Vous pouvez afficher ce document basculant vers l'option de l'éditeur JSON. Si ce document JSON dépasse la limite de taille d'une politique, l'éditeur visuel affiche un message d'erreur et ne vous permet pas de passer en revue et d'enregistrer votre politique. Pour afficher la limitation IAM relative à la taille d'une politique gérée, veuillez consulter Limites des caractères d'IAM et de STS.

Pour réduire la taille de votre politique dans l'éditeur visuel, modifiez votre politique ou déplacez les blocs d'autorisation dans une autre politique. Le message d'erreur inclut le nombre de caractères que votre document de politique contient, et vous pouvez utiliser ces informations pour vous aider à réduire la taille de votre politique.

Résolution des problèmes de services, d'actions ou de types de ressources non reconnus dans l'éditeur visuel

Lorsque vous créez ou modifiez une politique dans l'éditeur visuel, vous pouvez voir un avertissement indiquant que votre politique inclut un service, une action ou un type de ressource non reconnu.

> **Note**
>
> IAM vérifie les noms de services, les actions et les types de ressource pour les services qui prennent en charge les récapitulatifs de politique. Toutefois, le récapitulatif de votre politique

peut contenir une valeur de ressource ou une condition qui n'existe pas. Testez toujours vos stratégies avec le simulateur de stratégie.

Si votre stratégie inclut des services, des actions ou des types de ressource non reconnus, une des erreurs suivantes s'est produite :

- Service d'aperçu : les services en mode aperçu ne prennent pas en charge l'éditeur visuel. Si vous participez à l'aperçu, vous pouvez ignorer ce message d'avertissement et poursuivre, même si vous devez taper manuellement les ARN d'actions et de ressources pour finaliser votre politique. Vous pouvez également choisir l'option éditeur JSON afin de taper ou de coller un document de politique JSON.
- Service personnalisé : les services personnalisés ne prennent pas en charge l'éditeur visuel. Si vous utilisez un service personnalisé, vous pouvez ignorer ce message d'avertissement et poursuivre, même si vous devez taper manuellement les ARN d'actions et de ressources pour finaliser votre politique. Vous pouvez également choisir l'option éditeur JSON afin de taper ou de coller un document de politique JSON.
- Le service ne prend pas en charge l'éditeur visuel : si votre politique inclut un service disponible au public (GA) qui ne prend pas en charge l'éditeur visuel, vous pouvez ignorer ce message d'avertissement et poursuivre, même si vous devez taper manuellement les ARN d'actions et de ressources pour finaliser votre politique. Vous pouvez également choisir l'option éditeur JSON afin de taper ou de coller un document de politique JSON.

  Les services disponibles pour tous sont des services mis à disposition du public qui ne sont pas des services d'aperçu ou personnalisés. Si un service non reconnu est disponible pour tous et que son nom est orthographié correctement, il ne prend pas en charge l'éditeur visuel. Pour savoir comment demander la prise en charge de l'éditeur visuel ou du résumé de stratégie pour un service disponible pour tous (GA), consultez Le service ne prend pas en charge les récapitulatifs de politique IAM.
- L'action ne prend pas en charge l'éditeur visuel : si votre politique inclut un service pris en charge avec une action non prise en charge, vous pouvez ignorer ce message d'avertissement et poursuivre, même si vous devez taper manuellement les ARN d'actions et de ressources pour finaliser votre politique. Vous pouvez également choisir l'option éditeur JSON afin de taper ou de coller un document de politique JSON.

  Si votre politique contient un service pris en charge avec une action non prise en charge, le service ne prend pas en charge intégralement l'éditeur visuel. Pour savoir comment demander la prise en

charge de l'éditeur visuel ou du résumé de stratégie pour un service disponible pour tous (GA), consultez Le service ne prend pas en charge les récapitulatifs de politique IAM.

- Le type de ressource ne prend pas en charge l'éditeur visuel : si votre politique inclut une action prise en charge avec un type de ressource non pris en charge, vous pouvez ignorer le message d'avertissement et poursuivre. Toutefois, IAM ne peut pas confirmer que vous avez inclus des ressources pour l'ensemble de vos actions sélectionnées et vous pouvez consulter des avertissements supplémentaires.
- Faute de frappe : lorsque vous saisissez manuellement un service, une action ou une ressource dans l'éditeur visuel, vous pouvez parfois créer une politique qui inclut une faute de frappe. Pour éviter cela, nous vous recommandons d'utiliser l'éditeur visuel en sélectionnant dans la liste des services et des actions, puis de compléter la section des ressources en suivant les invites. Toutefois, si un service ne prend pas totalement en charge l'éditeur visuel, vous devrez peut-être taper manuellement certaines parties de votre politique.

  Si vous êtes sûr que votre politique ne contient aucune des erreurs ci-dessus, il se peut que votre politique contienne une faute de frappe. Recherchez d'éventuels noms de services, d'actions ou de types de ressource mal orthographiés. Par exemple, il se peut que vous ayez utilisé `s2` au lieu de `s3` et `ListMyBuckets` au lieu de `ListAllMyBuckets`. Une autre faute de frappe d'action courante est l'ajout de texte superflu dans les ARN, par exemple `arn:aws:s3: : :*` ou des points-virgules manquants dans les actions, comme `iam.CreateUser`. Vous pouvez évaluer une politique susceptible de contenir des fautes de frappe en choisissant Suivant afin de vérifier le résumé de la politique et de confirmer qu'elle contient les autorisations prévues.

Résolution des problèmes à l'aide des récapitulatifs de politique

Vous pouvez diagnostiquer et résoudre les problèmes liés aux résumés de politique.

Récapitulatif de politique manquant

La console IAM comprend des tables de récapitulatif de la politique qui présentent le niveau d'accès, les ressources et les conditions autorisées ou rejetées pour chaque service dans une politique. Les politiques sont résumées dans trois tables : récapitulatif de la politique, récapitulatif du service et récapitulatif de l'action. La table récapitulative de la politique comprend une liste des services et des résumés des autorisations définies par la politique choisie. Vous pouvez afficher le résumé de la politique pour toutes les politiques attachées à une entité sur la page Détails de la politique pour la politique en question. Vous pouvez afficher le récapitulatif des politiques gérées sur la page

Politiques. Si vous AWS ne parvenez pas à afficher le résumé d'une politique, le document de stratégie JSON s'affiche au lieu du résumé, et le message d'erreur suivant s'affiche :

Aucun récapitulatif pour cette politique ne peut être généré. Vous pouvez toujours afficher ou modifier le document de politique JSON.

Si votre politique ne comprend pas de récapitulatif, une des erreurs suivantes s'est produite :

- Élément de politique non pris en charge : IAM ne prend pas en charge la génération de récapitulatifs de politique pour les politiques comprenant l'un des éléments de politique suivants :
  - `Principal`
  - `NotPrincipal`
  - `NotResource`
- Aucune autorisation de politique : si une politique ne fournit pas d'autorisations effectives, le récapitulatif de politique ne peut pas être généré. Par exemple, si une politique contient une instruction unique avec l'élément `"NotAction": "*"`, cela signifie qu'elle accorde l'accès à toutes les actions, sauf « toutes les actions »(*). Cela signifie qu'elle accorde l'accès `Deny` ou `Allow` à rien.

  > **Note**
  >
  > Soyez prudent lorsque vous utilisez ces éléments de politique tels que `NotPrincipal`, `NotAction` et `NotResource`. Pour en savoir plus sur l'utilisation des éléments de la politique, veuillez consulter Références des éléments de politique JSON IAM.

  Vous pouvez créer une politique qui ne fournit pas d'autorisations effectives si vous fournissez des services et des ressources non appariés. Cela peut se produire si vous précisez des actions dans un service et des ressources provenant d'un autre service. Dans ce cas, le récapitulatif de la politique n'apparaît pas. La seule indication qu'un problème est survenu est que la colonne de ressource du récapitulatif contient une ressource provenant d'un service différent. Si cette colonne contient une ressource non appariée, recherchez d'éventuelles erreurs dans votre politique. Afin de mieux comprendre vos politiques, vous devez toujours les tester avec le simulateur de politique.

Le récapitulatif de politique inclut des services, des actions ou des types de ressource non reconnus

Dans la console IAM, si un récapitulatif de politique inclut un symbole d'avertissement ( ), la politique peut alors inclure un service, une action ou un type de ressource non reconnu. Pour en savoir plus sur les avertissements dans un récapitulatif de politique, veuillez consulter Récapitulatif de la politique (liste des services).

> **Note**
>
> IAM vérifie les noms de services, les actions et les types de ressource pour les services qui prennent en charge les récapitulatifs de politique. Toutefois, le récapitulatif de votre politique peut contenir une valeur de ressource ou une condition qui n'existe pas. Testez toujours vos stratégies avec le simulateur de stratégie.

Si votre stratégie inclut des services, des actions ou des types de ressource non reconnus, une des erreurs suivantes s'est produite :

- Service d'aperçu : les services en mode aperçu ne prennent pas en charge les récapitulatifs de politique.
- Service personnalisé : les services personnalisés ne prennent pas en charge les récapitulatifs de politique.
- Le service ne prend pas en charge les récapitulatifs : si votre politique contient un service disponible pour tous (GA) qui ne prend pas en charge les récapitulatifs de politique, le service figure dans la section Unrecognized services (Services non reconnus) de la table récapitulative de la politique. Les services disponibles pour tous sont des services mis à disposition du public qui ne sont pas des services d'aperçu ou personnalisés. Si un service non reconnu est disponible pour tous est que son nom est orthographié correctement, il ne prend pas en charge les récapitulatifs de politique IAM. Pour savoir comment demander la prise en charge du récapitulatif de politique d'un service disponible pour tous (GA), veuillez consulter Le service ne prend pas en charge les récapitulatifs de politique IAM.
- L'action ne prend pas en charge les récapitulatifs : si votre politique contient un service pris en charge avec une action non prise en charge, l'action figure alors dans la section Unrecognized actions (Actions non reconnues) de la table récapitulative du service. Pour en savoir plus sur

les avertissements dans un récapitulatif de service, consultez Récapitulatif du service (liste des actions).

- Le type de ressource ne prend pas en charge les récapitulatifs : si votre politique inclut une action prise en charge avec un type de ressource non pris en charge, la ressource figure dans la section Unrecognized resource types (Types de ressource non reconnus) de la table récapitulative du service. Pour en savoir plus sur les avertissements dans un récapitulatif de service, consultez Récapitulatif du service (liste des actions).
- Faute de frappe : AWS vérifie que le JSON est syntaxiquement correct et que la politique n'inclut pas de fautes de frappe ou d'autres erreurs dans le cadre de la validation de la politique.

> **Note**
>
> Selon les bonnes pratiques, nous vous recommandons d'utiliser IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sûres et fonctionnelles. Nous vous recommandons d'ouvrir vos politiques existantes, d'examiner et de résoudre toutes les recommandations de validation des politiques.

Le service ne prend pas en charge les récapitulatifs de politique IAM

Lorsqu'un service ou une action disponible pour tous (GA) n'est pas reconnu par les résumés de politique IAM ou l'éditeur visuel, il est possible que le service ne prenne pas en charge ces fonctionnalités. Les services disponibles pour tous sont des services mis à disposition du public qui ne sont pas des services d'aperçu ou personnalisés. Si un service non reconnu est disponible pour tous et que son nom est orthographié correctement, il ne prend pas en charge ces fonctionnalités. Si votre politique contient un service pris en charge avec une action non prise en charge, le service ne prend pas en charge intégralement les récapitulatifs de politique IAM.

Pour demander à ce qu'un service ajoute la prise en charge du résumé de politique IAM ou de l'éditeur visuel

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Recherchez la stratégie incluant le service non pris en charge :
   - Si la stratégie est une stratégie gérée, choisissez Stratégies dans le panneau de navigation. Dans la liste des stratégies, choisissez le nom de la stratégie à afficher.

- Si la stratégie est une stratégie en ligne attachée à l'utilisateur, choisissez Utilisateurs dans le panneau de navigation. Dans la liste des utilisateurs, choisissez le nom de l'utilisateur dont vous souhaitez afficher la politique. Dans le tableau des politiques dédiées à l'utilisateur, développez l'en-tête du récapitulatif de politique à afficher.

3. Sur le côté gauche du AWS Management Console pied de page, choisissez Feedback. Dans la case Commentaires pour IAM, tapez **`I request that the <ServiceName> service add support for IAM policy summaries and the visual editor`**. Si vous avez besoin de plusieurs services pour prendre en charge les récapitulatifs, saisissez **`I request that the <ServiceName1>, <ServiceName2>, and <ServiceName3> services add support for IAM policy summaries and the visual editor`**.

Pour demander que la prise en charge du récapitulatif de politique IAM d'une action manquante soit ajoutée à un service

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Recherchez la stratégie incluant le service non pris en charge :

   - Si la stratégie est une stratégie gérée, choisissez Stratégies dans le panneau de navigation. Dans la liste des stratégies, choisissez le nom de la stratégie à afficher.
   - Si la stratégie est une stratégie en ligne attachée à l'utilisateur, choisissez Utilisateurs dans le panneau de navigation. Dans la liste des utilisateurs, choisissez le nom de l'utilisateur dont vous souhaitez afficher la politique. Dans le tableau des politiques dédiées à l'utilisateur, sélectionnez le nom de la politique à afficher pour développer le récapitulatif de politique.

3. Dans le récapitulatif de la politique, sélectionnez le nom du service incluant une action non prise en charge.
4. Sur le côté gauche du AWS Management Console pied de page, choisissez Feedback. Dans la case Commentaires pour IAM, tapez **`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName> action`**. Si vous souhaitez signaler plusieurs actions non prises en charge, saisissez **`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName1>, <ActionName2>, and <ActionName3> actions`**.

Pour demander qu'un service différent contienne des actions manquantes, répétez les trois dernières étapes.

Ma politique n'accorde pas les autorisations escomptées

Pour attribuer des autorisations à un utilisateur, un groupe, un rôle ou une ressource, vous devez créer une politique, autrement dit un document qui définit les autorisations. Le document de politique inclut les éléments suivants :

- Effet : si la politique autorise ou refuse l'accès
- Action : la liste des actions qui sont autorisées ou rejetées par la politique
- Ressource : la liste des ressources sur lesquelles les actions peuvent se produire
- Condition (Facultatif) : les circonstances dans lesquelles la politique accorde l'autorisation

Pour en savoir plus sur ces éléments et d'autres éléments d'une politique, veuillez consulter Références des éléments de politique JSON IAM.

Pour accorder l'accès, votre politique doit définir une action avec une ressource prise en charge. Si votre politique inclut également une condition, cette condition doit inclure une clé de condition ou doit s'appliquer à l'action. Pour connaître les ressources qu'une action prend en charge, veuillez consulter la documentation AWS correspondant à votre service. Pour savoir quelles conditions sont prises en charge par une action, voir Actions, ressources et clés de condition pour les AWS services.

Pour savoir si votre politique définit une action, une ressource ou une condition qui n'accorde pas d'autorisations, vous pouvez afficher le récapitulatif de politique relatif à votre politique avec la console IAM à l'adresse https://console.aws.amazon.com/iam/. Vous pouvez utiliser les récapitulatifs de politiques afin d'identifier et de corriger les problèmes dans votre politique.

Plusieurs raisons peuvent expliquer l'incapacité d'un élément à accorder des autorisations bien que celles-ci soient définies dans la politique IAM :

- Une action est définie sans ressource applicable
- Une ressource est définie sans action applicable
- Une condition est définie sans action applicable

Pour voir des exemples de récapitulatifs de politiques incluant des avertissements, veuillez consulter the section called “Récapitulatif de la politique (liste des services)”.

Une action est définie sans ressource applicable

La politique ci-dessous définit toutes les actions `ec2:Describe*` et définit une ressource spécifique. Aucune des actions `ec2:Describe` n'est autorisée, car aucune de ces actions ne prend en charge les autorisations de niveau ressource. Les autorisations au niveau des ressources signifient que l'action prend en charge les ressources utilisant des noms ARN dans l'élément `Resource` de la politique. Si une action ne prend pas en charge les autorisations de niveau ressource, cette instruction de la politique doit utiliser le caractère générique (*) dans l'élément `Resource`. Pour savoir quels services prennent en charge les autorisations au niveau des ressources, veuillez consulter AWS services qui fonctionnent avec IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "arn:aws:ec2:us-east-2:ACCOUNT-ID:instance/*"
    }]
}
```

Cette politique ne fournit pas d'autorisations, et le récapitulatif de politique inclut l'erreur suivante :

```
This policy does not grant any permissions. To grant access, policies must
have an action that has an applicable resource or condition.
```

Pour corriger cette politique, vous devez utiliser le caractère générique * dans l'élément `Resource`.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
    }]
}
```

Une ressource est définie sans action applicable

La politique ci-dessous définit une ressource de compartiment Amazon S3, mais n'inclut pas d'action S3 pouvant être effectuée sur cette ressource. Cette politique accorde également un accès complet à toutes les CloudFront actions Amazon.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "cloudfront:*",
        "Resource": [
            "arn:aws:cloudfront:*",
            "arn:aws:s3:::examplebucket"
        ]
    }]
}
```

Cette politique fournit des autorisations pour toutes les CloudFront actions. Mais comme elle définit la ressource S3 `examplebucket` sans définir d'actions S3, le récapitulatif de politique inclut l'avertissement suivant :

`This policy defines some actions, resources, or conditions that do not provide permissions. To grant access, policies must have an action that has an applicable resource or condition.`

Pour corriger cette politique de façon à fournir des autorisations de compartiment S3, vous devez définir des actions S3 pouvant être exécutées sur une ressource du compartiment.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "cloudfront:*",
            "s3:CreateBucket",
            "s3:ListBucket*",
            "s3:PutBucket*",
            "s3:GetBucket*"
        ],
        "Resource": [
            "arn:aws:cloudfront:*",
            "arn:aws:s3:::examplebucket"
        ]
    }]
}
```

Sinon, pour corriger cette politique visant à fournir uniquement des CloudFront autorisations, supprimez la ressource S3.

Une condition est définie sans action applicable

La politique ci-dessous définit deux actions Amazon S3 pour toutes les ressources S3, si le préfixe S3 est égal à `custom` et l'ID de version à `1234`. Toutefois, la clé de condition `s3:VersionId` est utilisée pour le balisage de version d'objet et n'est pas pris en charge par les actions de compartiment définies. Pour savoir quelles conditions sont prises en charge par une action, consultez Actions, ressources et clés de condition pour les AWS services et suivez le lien vers la documentation du service pour les clés de condition.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

Cette politique fournit les autorisations pour l'action `s3:ListBucketVersions` et l'action `s3:ListBucket` si le nom du compartiment inclut le préfixe `custom`. Mais comme la condition `s3:VersionId` n'est prise en charge par aucune des actions définies, le récapitulatif de politique inclut l'erreur suivante :

```
This policy does not grant any permissions. To grant access, policies must
have an action that has an applicable resource or condition.
```

Pour corriger cette politique de façon à utiliser le balisage de version d'objet S3, vous devez définir une action S3 qui prend en charge la clé de condition `s3:VersionId`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket",
                "s3:GetObjectVersion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

Cette politique fournit des autorisations pour chaque action et condition de la politique. Toutefois, la politique ne fournit toujours pas d'autorisations, car il n'existe pas de cas où une même action correspond à deux conditions. À la place, vous devez créer deux instructions séparées : chacune comprendra uniquement les actions et les conditions auxquelles elles s'appliquent.

Pour corriger cette politique, créez deux instructions. La première instruction inclut les actions qui prennent en charge la condition `s3:prefix`, et la seconde instruction inclut les actions qui prennent en charge la condition `s3:VersionId`.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": "custom"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:GetObjectVersion",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:VersionId": "1234"
                }
            }
        }
    ]
}
```

Résolution des problèmes de gestion des politiques

Vous pouvez diagnostiquer et résoudre les problèmes liés à la gestion des politiques.

Attacher ou détacher une politique dans un compte IAM

Certaines politiques AWS gérées sont liées à un service. Ces politiques sont utilisées uniquement avec un rôle lié à un service pour ce service. Dans la console IAM, lorsque vous affichez la page Détails de la politique pour une politique, celle-ci contient une bannière signalant que la politique est liée à un service. Vous ne pouvez pas attacher cette politique à un utilisateur, un groupe ou un rôle au sein d'IAM. Lorsque vous créez un rôle lié à un service pour le service, cette politique est automatiquement attachée à votre nouveau rôle. Puisque la politique est obligatoire, vous ne pouvez pas la détacher du rôle lié au service.

Modification des politiques pour vos identités IAM en fonction de leur activité

Vous pouvez mettre à jour les politiques pour vos identités IAM (utilisateurs, groupes et rôles) en fonction de leur activité. Pour ce faire, consultez les événements de votre compte dans CloudTrail l'historique des événements. CloudTrail les journaux d'événements contiennent des informations détaillées sur les événements que vous pouvez utiliser pour modifier les autorisations de la politique. Il se peut qu'un utilisateur ou un rôle tente d'effectuer une action dans AWS et que cette demande soit refusée. Dans ce cas, vous pouvez voir si l'utilisateur ou le rôle doit avoir l'autorisation d'effectuer l'action. Si tel est le cas, vous pouvez ajouter l'action et même l'ARN de la ressource à laquelle il a tenté d'accéder avec sa politique. Sinon, si l'utilisateur ou le rôle dispose d'autorisations qu'il n'utilise pas, vous pouvez envisager de supprimer ces autorisations à partir de leur politique. Veillez à ce que vos politiques accordent le privilège le plus faible requis pour exécuter uniquement les actions nécessaires. Pour plus d'informations sur l'utilisation CloudTrail, consultez la section Affichage CloudTrail des événements dans la CloudTrail console dans le guide de AWS CloudTrail l'utilisateur.

Résolution des problèmes de documents de politique JSON

Vous pouvez diagnostiquer et résoudre les problèmes liés aux documents de politique JSON.

Valider vos politiques

Lorsque vous créez ou modifiez une politique JSON, IAM peut effectuer une validation de politique pour vous aider à créer une politique efficace. IAM identifie les erreurs de syntaxe JSON, tandis que IAM Access Analyzer fournit des vérifications de politique supplémentaires avec des recommandations pour vous aider à affiner vos politiques. Pour en savoir plus sur la validation de politiques, veuillez consulter Validation de politiques IAM. Pour en savoir plus sur les vérifications des politiques IAM Access Analyzer et les recommandations exploitables, veuillez consulter Validation de politique IAM Access Analyzer.

Je n'ai pas d'autorisations pour la validation de politique dans l'éditeur JSON

Dans le AWS Management Console, le message d'erreur suivant peut s'afficher si vous n'êtes pas autorisé à consulter les résultats de validation des politiques d'IAM Access Analyzer :

```
You need permissions. You do not have the permissions required to perform this operation. Ask your administrator to add permissions.
```

Pour corriger cette erreur, demandez à votre administrateur d'ajouter l'autorisation `access-analyzer:ValidatePolicy` pour vous.

Plusieurs objets de politique JSON

Une politique IAM doit inclure un seul objet JSON. Vous désignez un objet en le plaçant entre accolades { }. S'il est possible d'imbriquer d'autres objets au sein d'un objet JSON en incorporant des parenthèses { } supplémentaires dans la paire extérieure, une politique peut uniquement comporter une paire de parenthèses { } extérieure. L'exemple suivant est incorrect, car il comporte deux objets au premier niveau (indiqués en *rouge*) :

```
{
      "Version": "2012-10-17",
      "Statement":
      {
         "Effect":"Allow",
         "Action":"ec2:Describe*",
         "Resource":"*"
      }
   }
   {
     "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::my-bucket/*"
     }
   }
```

Toutefois, il est possible de rectifier l'exemple précédent à l'aide d'une syntaxe de politique appropriée. Au lieu d'utiliser deux objets de politique complets, avec chacun son propre élément `Statement`, vous pouvez combiner les deux blocs en un seul élément `Statement`. La valeur de l'élément `Statement` est un tableau de deux objets, comme illustré dans l'exemple suivant (marqué en gras) :

```
{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":" *"
        },
        {
          "Effect": "Allow",
```

```
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::my-bucket/*"
        }
      ]
    }
```

Plusieurs éléments de instruction JSON

Au premier abord, cette erreur peut sembler être une variante de celle de la section précédente. Toutefois, d'un point de vue syntaxique, il s'agit d'un type d'erreur différent. L'exemple suivant comporte un seul objet de politique, comme indiqué par la paire de parenthèses { } au premier niveau. Toutefois, cet objet contient deux éléments `Statement`.

Une politique IAM doit comporter un seul élément `Statement`, composé du nom (`Statement`) affiché à gauche d'un signe deux-points et suivi de sa valeur à droite. La valeur d'un élément `Statement` doit être un objet, indiqué par des accolades { }, contenant un élément `Effect`, un élément `Action` et un élément `Resource`. L'exemple suivant est incorrect, car il comporte deux éléments `Statement` dans l'objet de politique (marqué en *rouge*) :

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
      },
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::my-bucket/*"
      }
    }
```

Un objet de valeur peut être un tableau de plusieurs objets de valeur. Pour résoudre ce problème, combinez les deux éléments `Statement` en un même élément avec un tableau d'objets, comme illustré dans l'exemple suivant (marqué en gras) :

```
{
      "Version": "2012-10-17",
      "Statement": [    {
          "Effect": "Allow",
```

```
        "Action": "ec2:Describe*",
        "Resource":"*"
      },
      {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::my-bucket/*"
      }
   ]
  }
```

La valeur de l'élément `Statement` est un tableau d'objets. Dans cet exemple, le tableau se compose de deux objets, chaque objet étant à lui seul une valeur correcte pour un élément `Statement`. Chaque objet du tableau est séparé par des virgules.

Plusieurs éléments Effect, Action ou Resource dans un élément d'instruction JSON

Dans la partie valeur de la paire nom/valeur de `Statement`, l'objet doit comporter uniquement un élément `Effect`, un élément `Action` et un élément `Resource`. La politique suivante est incorrecte, car elle comporte deux éléments `Effect` dans l'objet de valeur de `Statement` :

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Effect": "Allow",
        "Action": "ec2:* ",
        "Resource": "*"
      }
    }
```

> **Note**
>
> Le moteur de politiques n'autorise pas ce type d'erreurs dans les politiques nouvelles ou modifiées. Il continue toutefois à autoriser les politiques enregistrées avant sa mise à jour. Les politiques existantes contenant l'erreur se comportent comme suit :
>
> - Plusieurs éléments `Effect` : seul le dernier élément `Effect` est pris en compte. Les autres sont ignorés.
> - Plusieurs éléments `Action` : tous les éléments `Action` sont combinés en interne et traités comme s'il s'agissait d'une seule liste.

- Plusieurs éléments `Resource` : tous les éléments `Resource` sont combinés en interne et traités comme s'il s'agissait d'une seule liste.

Le moteur de politique ne vous permet pas d'enregistrer une politique contenant des erreurs syntaxiques. Vous devez corriger les erreurs contenues dans la politique avant de l'enregistrer. Nous vous recommandons de consulter les recommandations de validation de politique appropriées pour vos politiques.

Dans chaque cas, la solution consiste à supprimer l'élément supplémentaire incorrect. Pour les éléments `Effect`, cela est très facile : si vous voulez que l'exemple précédent refuse des autorisations aux instances Amazon EC2, il vous suffit de supprimer la ligne `"Effect": "Allow",` de la politique, comme suit :

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Action": "ec2:* ",
        "Resource": "*"
      }
    }
```

Toutefois, lorsque l'élément dupliqué est `Action` ou `Resource`, la résolution peut s'avérer plus complexe. Vous voulez peut-être autoriser (ou refuser) des autorisations pour plusieurs actions ou vous souhaitez contrôler l'accès à plusieurs ressources. Par exemple, la stratégie suivante est incorrecte, car elle comporte plusieurs éléments `Resource` (marqués en *rouge*) :

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::my-bucket",
        "Resource": "arn:aws:s3:::my-bucket/*"
      }
    }
```

Chacun des éléments requis dans l'objet de valeur d'un élément `Statement` ne doit y figurer qu'une seule fois. La solution consiste à placer chaque valeur dans un tableau. L'exemple suivant illustre cette méthode : les deux éléments Resource sont inclus dans un même élément `Resource` avec un tableau comme objet de valeur (marqué en gras) :

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": [
          "arn:aws:s3:::my-bucket",
          "arn:aws:s3:::my-bucket/*"
        ]
      }
    }
```

Élément de version JSON manquant

Un élément de politique `Version` est différent d'une version de politique. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. En revanche, une version de politique est créée lorsque vous apportez des modifications à une politique gérée par le client dans IAM. La politique modifiée ne remplace pas la politique existante. À la place, IAM crée une nouvelle version de la politique gérée. Pour en savoir plus sur l'élément de politique `Version`, consultez Éléments de politique JSON IAM : Version. Pour en savoir plus sur les versions de politiques, consultez the section called “Gestion des versions des politiques IAM”.

À mesure que les AWS fonctionnalités évoluent, de nouvelles fonctionnalités sont ajoutées aux politiques IAM pour prendre en charge ces fonctionnalités. Parfois, la mise à jour de la syntaxe de politique inclut un nouveau numéro de version. Si vous utilisez les nouvelles fonctions de la syntaxe de politique dans votre politique, vous devez indiquer le numéro de la version utilisée au moteur d'analyse de politique. Le numéro de version de politique par défaut est « 2008-10-17 ». Si vous voulez utiliser une fonction de politique plus récente, vous devez spécifier le numéro de version prenant en charge cette fonction. Nous vous recommandons de toujours inclure le numéro de version de syntaxe de politique le plus récent, à savoir actuellement `"Version": "2012-10-17"`. Par exemple, la politique suivante est incorrecte, car elle utilise une variable de politique `${...}` dans l'ARN d'une ressource. Mais elle ne parvient pas à spécifier une version de syntaxe de politique prenant en charge les variables de politique (marquées en *`rouge`*) :

```
{
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

L'ajout d'un élément `Version` au début de la politique avec la valeur `2012-10-17`, la première version de l'API IAM qui prend en charge les variables de politique, permet de corriger ce problème (marqué en gras) :

```
{
  "Version": "2012-10-17",
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

Résolution des problèmes liés aux clés de sécurité FIDO

Utilisez ces informations pour identifier et résoudre les problèmes courants que vous pouvez rencontrer lors de l'utilisation des clés de sécurité FIDO2.

Rubriques

- Je ne peux pas activer ma clé de sécurité FIDO
- Je ne peux pas me connecter à l'aide de ma clé de sécurité FIDO
- J'ai perdu ou cassé ma clé de sécurité FIDO
- Autres problèmes

Je ne peux pas activer ma clé de sécurité FIDO

Consultez les solutions suivantes en fonction de votre statut (utilisateur IAM ou administrateur système).

Utilisateurs IAM

Si vous ne pouvez pas activer votre clé de sécurité FIDO, vérifiez les points suivants :

- Utilisez-vous une configuration prise en charge ?

  Pour plus d'informations sur les appareils et les navigateurs que vous pouvez utiliser avec WebAuthn et AWS, voirConfigurations prises en charge pour l'utilisation des clés d'accès et des clés de sécurité.

- Utilisez-vous Mozilla Firefox ?

  Les versions actuelles de Firefox sont prises WebAuthn en charge par défaut. Pour activer la prise WebAuthn en charge de Firefox, procédez comme suit :

  1. Dans la barre d'adresse Firefox, saisissez **`about:config`**.
  2. Dans la barre de recherche de l'écran qui s'ouvre, saisissez **`webauthn`**.
  3. Choisissez security.webauth.webauthn et remplacez sa valeur par true (vrai).

- Utilisez-vous des plug-ins de navigateur ?

  AWS ne prend pas en charge l'utilisation de plugins pour ajouter le support WebAuthn du navigateur. Utilisez plutôt un navigateur qui offre une prise en charge native de la WebAuthn norme.

  Même si vous utilisez un navigateur compatible, il se peut que vous disposiez d'un plugin incompatible avec WebAuthn. Un plug-in incompatible peut vous empêcher d'activer et d'utiliser votre clé de sécurité conforme à FIDO. Vous devez désactiver les plug-ins qui pourraient être incompatibles et redémarrer votre navigateur. Réessayez ensuite d'activer la clé de sécurité FIDO.

- Disposez-vous des autorisations appropriées ?

  Si vous ne rencontrez pas l'un des problèmes de compatibilité ci-dessus, vous n'avez peut-être pas les autorisations appropriées. Contactez votre administrateur système.

Administrateurs système

Si vous êtes administrateur et que vos utilisateurs IAM ne peuvent pas activer leurs clés de sécurité FIDO bien qu'ils utilisent une configuration prise en charge, assurez-vous qu'ils disposent des autorisations appropriées. Pour voir un exemple détaillé, veuillez consulter Didacticiel IAM : permettre aux utilisateurs de gérer leurs informations d'identification et leurs paramètres MFA.

Je ne peux pas me connecter à l'aide de ma clé de sécurité FIDO

Si vous êtes un utilisateur IAM et que vous ne pouvez pas vous connecter à l' AWS Management Console aide de votre clé de sécurité FIDO, consultez d'abord. Configurations prises en charge pour l'utilisation des clés d'accès et des clés de sécurité Si vous utilisez une configuration prise en charge mais que vous ne pouvez pas vous connecter, contactez votre administrateur système pour obtenir de l'aide.

J'ai perdu ou cassé ma clé de sécurité FIDO

Jusqu'à huit dispositifs MFA de n'importe quelle combinaison de types MFA actuellement pris en charge peuvent être attribués à un utilisateur. Avec plusieurs dispositifs MFA, vous n'en avez besoin que d'un seul pour vous connecter à la AWS Management Console. Le remplacement d'une clé de sécurité FIDO est semblable à celui d'un jeton TOTP matériel. Pour plus d'informations sur la procédure à suivre en cas de perte ou de casse d'un appareil MFA, veuillez consulter Que faire si un dispositif MFA est perdu ou cesse de fonctionner ?.

Autres problèmes

Si vous rencontrez un problème avec les clés de sécurité FIDO qui n'est pas traité ici, effectuez l'une des actions suivantes :

- Utilisateurs IAM : contactez votre administrateur système.
- Utilisateurs racine Compte AWS : contactez le Support AWS.

Résolution des problèmes liés aux rôles IAM

Utilisez ces informations pour identifier et résoudre les problèmes courants que vous pouvez rencontrer lors de l'utilisation des rôles IAM.

Rubriques

- Je ne parviens pas à endosser un rôle
- Un nouveau rôle est apparu dans mon compte AWS
- Je ne parviens pas à modifier ou supprimer un rôle dans mon Compte AWS
- Je ne suis pas autorisé à exécuter : iam : PassRole
- Pourquoi ne puis-je pas endosser un rôle avec une session de 12 heures ? (AWS CLI, AWS API)

- Je reçois une erreur lorsque j'essaie de changer de rôle dans la console IAM
- Mon rôle dispose d'une politique qui me permet d'effectuer une action, mais j'obtiens « Accès refusé »
- Le service n'a pas créé la version de politique par défaut du rôle
- Il n'existe aucun cas d'utilisation pour un rôle de service dans la console

Je ne parviens pas à endosser un rôle

Vérifiez les éléments suivants :

- Pour permettre aux utilisateurs d'assumer à nouveau le rôle actuel au cours d'une session de rôle, spécifiez l'ARN du rôle ou l' Compte AWS ARN comme principal dans la politique de confiance des rôles. Services AWS qui fournissent des ressources de calcul telles qu'Amazon EC2, Amazon ECS, Amazon EKS et Lambda fournissent des informations d'identification temporaires et mettent automatiquement à jour ces informations d'identification. Cela garantit que vous disposez toujours d'un ensemble d'informations d'identification valide. Pour ces services, il n'est pas nécessaire d'endosser à nouveau le rôle actuel pour obtenir des informations d'identification temporaires. Toutefois, si vous avez l'intention de transférer des balises de session ou une politique de session, vous devez endosser à nouveau le rôle actuel. Pour savoir comment modifier une politique d'approbation des rôles afin d'ajouter l'ARN ou Compte AWS l'ARN du rôle principal, consultezModification d'une politique d'approbation de rôle (console).
- Lorsque vous assumez un rôle en utilisant le AWS Management Console, assurez-vous d'utiliser le nom exact de votre rôle. Les noms de rôle sont sensibles à la casse lorsque vous endossez un rôle.
- Lorsque vous assumez un rôle à l'aide de l' AWS STS API ou AWS CLI, assurez-vous d'utiliser le nom exact de votre rôle dans l'ARN. Les noms de rôle sont sensibles à la casse lorsque vous endossez un rôle.
- Vérifiez que votre politique IAM vous accorde l'autorisation d'appeler `sts:AssumeRole` pour le rôle que vous voulez endosser. L'élément `Action` de votre politique IAM doit vous autoriser à appeler l'action `AssumeRole`. En outre, l'élément `Resource` de votre politique IAM doit spécifier le rôle que vous voulez endosser. Par exemple, l'élément `Resource` peut spécifier un rôle en utilisant son ARN (Amazon Resource Name) ou à l'aide d'un caractère générique (*). Par exemple, au moins une politique s'appliquant à vous doit accorder des autorisations similaires à ce qui suit :

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
```

```
    "Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
```

- Vérifiez que votre identité IAM est balisée avec les balises que la politique IAM requiert. Par exemple, dans la politique d'autorisations suivante, l'élément `Condition` exige que vous, en tant que principal demandant à endosser le rôle, ayez une balise spécifique. Vous devez être balisé avec `department = HR` ou `department = CS`. Dans le cas contraire, vous ne pouvez pas endosser le rôle. Pour en savoir plus sur le balisage des utilisateurs et rôles IAM, veuillez consulter the section called “Balisage des ressources IAM”.

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "*",
    "Condition": {"StringEquals": {"aws:PrincipalTag/department": [
            "HR",
            "CS"
        ]}}
```

- Assurez-vous que vous respectez toutes les conditions spécifiées dans la politique d'approbation du rôle. Une `Condition` peut spécifier une date d'expiration, un ID externe, ou exiger qu'une demande provienne uniquement d'adresses IP spécifiques. Prenons l'exemple suivant. Si la date actuelle se situe après la date spécifiée, la politique ne correspond jamais et, par conséquent, elle ne peut par vous accorder l'autorisation d'endosser le rôle.

```
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
    "Condition": {
        "DateLessThan" : {
            "aws:CurrentTime" : "2016-05-01T12:00:00Z"
        }
    }
```

- Vérifiez que l'entité Compte AWS à partir de laquelle vous appelez `AssumeRole` est une entité de confiance pour le rôle que vous assumez. Les entités approuvées sont définies en tant que `Principal` dans la politique d'approbation d'un rôle. L'exemple suivant est une politique de confiance attachée au rôle que vous voulez endosser. Dans cet exemple, l'ID de compte avec l'utilisateur IAM avec lequel vous vous êtes connecté doit être 123456789012. Si votre numéro de compte n'est pas répertorié dans l'élément `Principal` de la politique de confiance du rôle, vous ne pouvez pas endosser le rôle. Peu importent les autorisations qui vous sont accordées dans les politiques d'accès. Notez que l'exemple de politique limite les autorisations aux actions

susceptibles de se produire entre le 1er juillet 2017 et le 31 décembre 2017 (UTC), inclus. Si vous vous connectez avant ou après ces dates, la politique de correspond pas et vous ne pouvez pas endosser le rôle.

```
    "Effect": "Allow",
    "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
    "Action": "sts:AssumeRole",
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
    }
```

- Identité source : les administrateurs peuvent configurer les rôles pour exiger que les identités transmettent une chaîne personnalisée identifiant la personne ou l'application qui exécute des actions AWS, appelée identité source. Vérifiez si le rôle endossé exige qu'une identité source soit définie. Pour de plus amples informations sur l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

Un nouveau rôle est apparu dans mon compte AWS

Certains AWS services nécessitent que vous utilisiez un type unique de rôle de service directement lié au service. Ce rôle lié à un service est prédéfini par le service et comprend toutes les autorisations nécessaires au service. La configuration d'un service est ainsi simplifiée, étant donné que vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Pour obtenir des informations générales sur les rôles liés à un service, veuillez consulter Utilisation des rôles liés à un service.

Vous utilisez peut être déjà un service lorsqu'il commence à prendre en charge des rôles de service liés à un service. Dans ce cas, il est possible que vous receviez un e-mail vous informant d'un nouveau rôle dans votre compte. Ce rôle comprend toutes les autorisations dont le service a besoin pour effectuer des actions en votre nom. Aucune action de votre part n'est requise pour prendre ce rôle en charge. Cependant, vous ne devez pas supprimer le rôle de votre compte. En effet, la suppression risquerait de supprimer des autorisations dont le service a besoin pour accéder aux ressources AWS . Pour afficher les rôles liés à un service de votre compte, allez sur la page IAM Roles (Rôles IAM) de la console IAM. Les rôles liés à un service s'affichent avec (Rôle lié à un service) mentionné dans la colonne Entités de confiance du tableau.

Pour plus d'informations concernant la prise en charge par les services des rôles liés à un service, reportez-vous à la rubrique AWS services qui fonctionnent avec IAM et recherchez les services

affichant Oui dans la colonne Rôle lié à un service. Pour plus d'informations sur l'utilisation du rôle lié à un service, sélectionnez le lien Oui.

Je ne parviens pas à modifier ou supprimer un rôle dans mon Compte AWS

Vous ne pouvez pas supprimer ni modifier les autorisations d'un rôle lié à un service dans IAM. Ces rôles comportent des approbations et des autorisations prédéfinies requises par le service afin d'exécuter des actions en votre nom. Vous pouvez utiliser la console IAM ou l'API pour modifier uniquement la description d'un rôle lié à un service. AWS CLI Pour afficher les rôles liés à un service de votre compte, rendez-vous sur la page IAM Roles (Rôles IAM) de la console. Les rôles liés à un service s'affichent avec (Rôle lié à un service) mentionné dans la colonne Entités de confiance du tableau. Une bannière sur la page Récapitulatif d'un rôle indique aussi que le rôle est lié à un service. Vous pouvez gérer et supprimer ces rôles uniquement via le service lié, si celui-ci prend en charge l'action. Soyez vigilant lorsque vous modifiez ou supprimez un rôle lié à un service car cela peut supprimer des autorisations requises par le service pour accéder aux ressources AWS .

Pour plus d'informations concernant la prise en charge par les services des rôles liés à un service, reportez-vous à la rubrique AWS services qui fonctionnent avec IAM et recherchez les services affichant Oui dans la colonne Rôle lié à un service.

Je ne suis pas autorisé à exécuter : iam : PassRole

Lorsque vous créez un rôle lié à un service, vous devez être autorisé à transmettre ce rôle au service. Certains services créent automatiquement un rôle lié à un service dans votre compte lorsque vous effectuez une action dans ce service. Par exemple, Amazon EC2 Auto Scaling crée automatiquement le rôle lié au service `AWSServiceRoleForAutoScaling` la première fois que vous créez un groupe Auto Scaling. Si vous essayez de créer un groupe Auto Scaling sans l'autorisation `PassRole`, vous recevez le message d'erreur suivant :

```
ClientError: An error occurred (AccessDenied) when calling the
PutLifecycleHook operation: User: arn:aws:sts::111122223333:assumed-role/
Testrole/Diego is not authorized to perform: iam:PassRole on resource:
arn:aws:iam::111122223333:role/aws-service-role/autoscaling.amazonaws.com/
AWSServiceRoleForAutoScaling
```

Pour corriger cette erreur, demandez à votre administrateur d'ajouter l'autorisation `iam:PassRole` pour vous.

Pour savoir quels services prennent en charge les rôles liés à un service, veuillez consulter AWS services qui fonctionnent avec IAM. Pour savoir si un service crée automatiquement un rôle lié à un service, sélectionnez le lien Oui pour afficher la documentation du rôle lié à ce service.

Pourquoi ne puis-je pas endosser un rôle avec une session de 12 heures ? (AWS CLI, AWS API)

Lorsque vous utilisez les opérations de l' AWS STS `AssumeRole*`API ou de la `assume-role*` CLI pour assumer un rôle, vous pouvez spécifier une valeur pour le `DurationSeconds` paramètre. Vous pouvez spécifier une valeur comprise entre 900 secondes (15 minutes) et la valeur de durée de session maximale définie pour le rôle. Si vous spécifiez une valeur supérieure à ce paramètre, l'opération échoue. La valeur maximale de ce paramètre est de 12 heures. Par exemple, si vous spécifiez une durée de session de 12 heures, mais que votre administrateur a défini la durée de session maximale à 6 heures, votre opération échoue. Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Affichage du paramètre de durée de session maximale pour un rôle.

Si vous utilisez la création de chaînes de rôles (en utilisant un rôle pour endosser un second rôle), votre session est limitée à une heure maximum. Si vous utilisez ensuite le paramètre `DurationSeconds` pour fournir une valeur supérieure à une heure, l'opération échoue.

Je reçois une erreur lorsque j'essaie de changer de rôle dans la console IAM

Les informations que vous entrez sur la page Changer de rôle doivent correspondre à celles du rôle. Sinon, l'opération échoue et vous recevez l'erreur suivante :

```
Invalid information in one or more fields. Check your information or
contact your administrator.
```

Si vous recevez cette erreur, confirmez que les informations suivantes sont correctes :

- Identifiant de compte ou alias — L' Compte AWS identifiant est un numéro à 12 chiffres. Votre compte peut avoir un alias, qui est un identifiant convivial tel que le nom de votre entreprise qui peut être utilisé à la place de votre Compte AWS identifiant. Vous pouvez utiliser l'ID de compte ou l'alias dans ce champ.
- Role name (Nom de rôle) : les noms de rôle sont sensibles à la casse. L'ID du compte et le nom du rôle doivent correspondre à ce qui est configuré pour le rôle.

Si vous continuez à recevoir un message d'erreur, contactez votre administrateur pour vérifier les informations précédentes. La politique d'approbation de rôle ou la politique utilisateur IAM peut limiter votre accès. Votre administrateur peut vérifier les autorisations pour ces politiques.

Mon rôle dispose d'une politique qui me permet d'effectuer une action, mais j'obtiens « Accès refusé »

Votre session peut être limitée par les politiques de session. Lorsque vous demandez des informations d'identification de sécurité temporaires par le biais d'un programme AWS STS, vous pouvez éventuellement transmettre des politiques de session en ligne ou gérées. Les politiques de session sont des politiques avancées que vous transmettez en tant que paramètre lorsque vous créez par programmation une session d'informations d'identification temporaires pour un rôle. Vous pouvez transmettre un seul document de politique de session en ligne JSON à l'aide du paramètre `Policy`. Vous pouvez utiliser le paramètre `PolicyArns` pour spécifier jusqu'à 10 politiques de session gérées. Les autorisations de la session obtenues sont une combinaison des politiques basées sur l'identité du rôle et des politiques de session. Sinon, si votre administrateur ou un programme personnalisé vous fournit des informations d'identification temporaires, ils peuvent avoir inclus une politique de session pour limiter votre accès.

Le service n'a pas créé la version de politique par défaut du rôle

Un rôle de service est un rôle qu'un service endosse pour effectuer des actions dans votre compte en votre nom. Lorsque vous configurez certains environnements de AWS service, vous devez définir le rôle que le service doit assumer. Dans certains cas, le service crée le rôle de service et sa politique dans IAM pour vous. Bien que vous puissiez modifier ou supprimer le rôle de service et sa politique depuis IAM, AWS ne le recommande pas. Le rôle et la politique sont destinés à être utilisés uniquement par ce service. Si vous modifiez la politique et configurez un autre environnement, lorsque le service tente d'utiliser le même rôle et la même politique, l'opération peut échouer.

Par exemple, lorsque vous l'utilisez AWS CodeBuild pour la première fois, le service crée un rôle nommé`codebuild-RWBCore-service-role`. Ce rôle de service utilise la politique nommée `codebuild-RWBCore-managed-policy`. Si vous modifiez la politique, elle crée une nouvelle version et enregistre cette version en tant que version par défaut. Si vous effectuez une opération ultérieure dans AWS CodeBuild, le service peut essayer de mettre à jour la politique. Si c'est le cas, vous recevez l'erreur suivante :

```
codebuild.amazon.com did not create the default version (V2) of the
codebuild-RWBCore-managed-policy policy that is attached to the codebuild-
```

`RWBCore-service-role role. To continue, detach the policy from any other identities and then delete the policy and the role.`

Si vous recevez cette erreur, vous devez apporter des modifications dans IAM avant de pouvoir poursuivre votre opération de service. Tout d'abord, définissez la version de la politique par défaut sur V1 et réessayez l'opération. Si V1 a déjà été supprimé, ou si le choix de V1 ne fonctionne pas, nettoyez et supprimez la politique et le rôle existants.

Pour de plus amples informations sur la modification des politiques gérées, veuillez consulter Modification de politiques gérées par le client (console). Pour plus d'informations sur les versions de politique, veuillez consulter Gestion des versions des politiques IAM.

Pour supprimer un rôle de service et sa politique

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Policies (Politiques).
3. Dans la liste des politiques, sélectionnez le nom de la politique à supprimer.
4. Choisissez l'onglet Entités attachées pour afficher les utilisateurs, groupes ou rôles IAM qui utilisent cette politique. Si l'une de ces identités utilise la politique, effectuez les tâches suivantes :

    a. Créez une nouvelle politique gérée avec les autorisations nécessaires. Pour vous assurer que les identités disposent des mêmes autorisations avant et après vos actions, copiez le document de politique JSON à partir de la politique existante. Créez ensuite la nouvelle politique gérée et collez le document JSON comme décrit dans la section Création de politiques à l'aide de l'éditeur JSON.

    b. Pour chaque identité affectée, joignez la nouvelle politique, puis détachez l'ancienne. Pour plus d'informations, consultez Ajout et suppression d'autorisations basées sur l'identité IAM.
5. Dans le panneau de navigation, sélectionnez Roles (Rôles).
6. Dans la liste des rôles, sélectionnez le nom du rôle à supprimer.
7. Cliquez sur l'onglet Relations d'approbation pour afficher les entités qui peuvent endosser le rôle. Si une entité autre que le service est répertoriée, effectuez les tâches suivantes :

    a. Créez un nouveau rôle qui approuve ces entités.

    b. Politique que vous avez créée à l'étape précédente. Si vous avez ignoré cette étape, créez la nouvelle politique gérée maintenant.

c. Avisez toute personne qui endossait le rôle qu'elle ne peut plus le faire. Donnez-lui des informations sur la façon d'endosser le nouveau rôle et d'avoir les mêmes autorisations.

8. Supprimez la politique.
9. Supprimez le rôle.

Il n'existe aucun cas d'utilisation pour un rôle de service dans la console

Certains services exigent que vous créiez manuellement un rôle de service pour accorder au service les autorisations nécessaires pour effectuer des actions en votre nom. Si le service n'est pas répertorié dans la console IAM, vous devez répertorier manuellement le service en tant que principal de confiance. Si la documentation du service ou de la fonctionnalité que vous utilisez ne contient pas d'instructions pour intégrer le service à la liste des principaux de confiance, soumettez des commentaires pour la page.

Pour créer manuellement un rôle de service, vous devez connaître le principal de service du service qui endossera le rôle. Un principal de service est un identifiant utilisé pour accorder des autorisations à un service. Le principal de service est défini par le service.

Vous pouvez trouver le principal de service pour certains services en vérifiant les éléments suivants :

1. Ouvrir AWS services qui fonctionnent avec IAM.
2. Vérifiez si le service indique Oui dans la colonne Rôles liés à un service .
3. Choisissez le lien Oui pour consulter la documentation du rôle lié à ce service.
4. Recherchez la section Autorisations de rôles liés à un service correspondant à ce service pour afficher le principal du service.

Vous pouvez créer manuellement un rôle de service à l'aide de commandes AWS CLI ou d'opérations d'API AWS. Pour créer manuellement un rôle de service à l'aide de la console IAM, effectuez les tâches suivantes :

1. Créez un rôle IAM à l'aide de votre ID de compte. N'attachez pas de politique et n'accordez aucune autorisation. Pour plus de détails, consultez Création d'un rôle pour la délégation d'autorisations à un utilisateur IAM..
2. Ouvrez le rôle et modifiez la relation d'approbation. Au lieu d'approuver le compte, le rôle doit approuver le service. Par exemple, mettez à jour l'élément `Principal` suivant :

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

Modifiez le principal par la valeur de votre service, par exemple, IAM.

```
"Principal": { "Service": "iam.amazonaws.com" }
```

3. Ajoutez les autorisations requises par le service en attachant des politiques d'autorisations au rôle.
4. Revenez au service qui nécessite les autorisations et utilisez la méthode décrite dans la documentation pour informer le service du nouveau rôle de service.

Dépannage d'IAM et Amazon EC2

Utilisez ces informations pour résoudre et corriger les problèmes d'accès refusé ou autres que vous pouvez rencontrer lors de l'utilisation d'Amazon EC2 et IAM.

Rubriques

- Lors de la tentative de lancement de l'instance, je ne vois pas le rôle attendu dans la liste de Rôles IAM de la console Amazon EC2.
- Les informations d'identification sur mon instance concernent le mauvais rôle
- Quand je tente d'appeler AddRoleToInstanceProfile, je reçois un message d'erreur AccessDenied.
- Amazon EC2 : quand je tente de lancer l'instance avec un rôle, je reçois un message d'erreur AccessDenied
- Je ne parviens pas à accéder aux informations d'identification de sécurité temporaires sur mon instance EC2
- Que signifient les erreurs dans le document info de la sous-arborescence IAM ?

Lors de la tentative de lancement de l'instance, je ne vois pas le rôle attendu dans la liste de Rôles IAM de la console Amazon EC2.

Vérifiez les éléments suivants :

- Si vous êtes connecté en tant qu'utilisateur IAM, vérifiez que vous avez l'autorisation d'appeler `ListInstanceProfiles`. Pour plus d'informations sur les autorisations nécessaires pour utiliser ces rôles, veuillez consulter « Autorisations requises pour utiliser les rôles avec Amazon EC2 »

dans Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2. Pour de plus amples informations sur l'ajout d'autorisations à un utilisateur, veuillez consulter Gestion des politiques IAM.

Si vous ne pouvez pas modifier vos propres autorisations, vous devez contacter un administrateur qui peut utiliser IAM pour mettre à jour vos autorisations.

- Si vous avez créé un rôle à l'aide de la CLI ou l'API IAM, vérifiez que vous avez créé un profil d'instance et ajouté le rôle à celui-ci. De plus, si vous nommez votre rôle et votre profil d'instance différemment, le nom correct du rôle ne s'affichera pas dans la liste des rôles IAM de la console Amazon EC2. La liste des Rôles IAM de la console Amazon EC2 répertorie les noms des profils d'instance, pas les noms des rôles. Vous devez sélectionner le nom du profil d'instance qui contient le rôle choisi. Pour plus informations sur les profils d'instance, veuillez consulter Utilisation de profils d'instance.

> **Note**
>
> Si vous utilisez la console IAM pour créer des rôles, vous n'avez pas besoin d'utiliser des profils d'instance. Pour chaque rôle que vous créez dans la console IAM, un profil d'instance est créé avec un nom identique à celui du rôle, et le rôle est ajouté automatiquement à ce profil d'instance. Un profil d'instance peut contenir un rôle IAM uniquement et cette limite ne peut pas être augmentée.

Les informations d'identification sur mon instance concernent le mauvais rôle

Le rôle dans le profil d'instance a peut-être été remplacé récemment. Si tel est le cas, votre application attendra la prochaine rotation des informations d'identification planifiée automatiquement pour que les informations d'identification de votre rôle deviennent disponibles.

Pour forcer la modification, vous devez dissocier le profil d'instance, puis associer le profil d'instance, ou vous pouvez arrêter votre instance, puis la redémarrer.

Quand je tente d'appeler **`AddRoleToInstanceProfile`**, je reçois un message d'erreur **`AccessDenied`**.

Si vous faites des demandes en tant qu'utilisateur IAM, vérifiez que vous avez les autorisation suivantes :

- `iam:AddRoleToInstanceProfile` avec la ressource correspondant à l'ARN du profil d'instance (par exemple, `arn:aws:iam::999999999999:instance-profile/ExampleInstanceProfile`).

Pour plus d'informations sur les autorisations nécessaires pour utiliser ces rôles, veuillez consulter « Comment bénéficier du service ? ». dans Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2. Pour de plus amples informations sur l'ajout d'autorisations à un utilisateur, veuillez consulter Gestion des politiques IAM.

Amazon EC2 : quand je tente de lancer l'instance avec un rôle, je reçois un message d'erreur **`AccessDenied`**

Vérifiez les éléments suivants :

- Lancez une instance sans profil d'instance. Cela vous permettra de vérifier que le problème se limite aux rôles IAM dans les instances Amazon EC2.
- Si vous faites des demandes en tant qu'utilisateur IAM, vérifiez que vous avez les autorisation suivantes :
  - `ec2:RunInstances` avec une ressource générique (« * »)
  - `iam:PassRole` avec la ressource correspondant à l'ARN du rôle (par exemple, `arn:aws:iam::999999999999:role/ExampleRoleName`)
- Appelez l'action IAM `GetInstanceProfile` pour vous assurer que vous utilisez un nom de profil d'instance valide ou un ARN de profil d'instance valide. Pour plus d'informations, veuillez consulter Utilisation de rôles IAM avec les instances Amazon EC2.
- Appelez l'action `GetInstanceProfile` IAM pour vous assurer que le profil d'instance dispose d'un rôle. Les profils d'instance vides échoueront et renverront l'erreur `AccessDenied`. Pour plus d'informations sur la création d'un rôle, veuillez consulter Création de rôles IAM.

Pour plus d'informations sur les autorisations nécessaires pour utiliser ces rôles, veuillez consulter « Comment bénéficier du service ? ». dans Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2. Pour de plus amples informations sur l'ajout d'autorisations à un utilisateur, veuillez consulter Gestion des politiques IAM.

Je ne parviens pas à accéder aux informations d'identification de sécurité temporaires sur mon instance EC2

Pour accéder aux informations d'identification de sécurité temporaires sur votre instance EC2, vous devez d'abord utiliser la console IAM pour créer un rôle. Ensuite, vous lancez une instance EC2 qui utilise ce rôle et vous examinez l'instance en cours d'exécution. Pour de plus amples informations, veuillez consulter Comment démarrer avec ce service ? dans Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2.

Si vous ne parvenez toujours pas à accéder à vos informations d'identification de sécurité temporaires sur votre instance EC2, vérifiez les points suivants :

- Pouvez-vous accéder à une autre partie du service des métadonnées d'instance (IMDS) ? Si ce n'est pas le cas, vérifiez qu'aucune règle de pare-feu ne bloque l'accès aux demandes envoyées à l'IMDS.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/
hostname; echo
```

- La sous-arborescence `iam` de l'IMDS existe-elle ? Si ce n'est pas le cas, vérifiez que votre instance dispose d'un profil d'instance IAM associé en appelant l'opération d'API EC2 `DescribeInstances` ou en utilisant la commande de la CLI `aws ec2 describe-instances`.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam;
 echo
```

- Consultez le document `info` dans la sous-arborescence IAM pour rechercher une erreur. S'il existe une erreur, veuillez consulter Que signifient les erreurs dans le document `info` de la sous-arborescence IAM ? pour plus d'informations.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam/
info; echo
```

Que signifient les erreurs dans le document **`info`** de la sous-arborescence IAM ?

Le document **`iam/info`** indique **`"Code":"InstanceProfileNotFound"`**

Votre profil d'instance IAM a été supprimé et Amazon EC2 ne peut plus fournir d'informations d'identification à votre instance. Vous devez attacher un profil d'instance valide à votre instance Amazon EC2.

S'il existe un profil d'instance avec le même nom, vérifiez que celui-ci n'a pas été supprimé et qu'un autre n'a pas été créé avec le même nom :

1. Appelez l'opération `GetInstanceProfile` IAM pour obtenir l'ID de profil d'instance `InstanceProfileId`.
2. Appelez l'opération `DescribeInstances` Amazon EC2 pour obtenir l'ID de profil d'instance `IamInstanceProfileId`.
3. Vérifiez que l'ID `InstanceProfileId` de l'opération IAM correspond à l'ID `IamInstanceProfileId` de l'opération Amazon EC2.

Si les ID sont différents, le profil d'instance attaché à vos instances n'est plus valide. Vous devez attacher un profil d'instance valide à l'instance.

Le document **`iam/info`** indique un succès, mais renvoie le message **`"Message":"Instance Profile does not contain a role..."`**

Le rôle a été supprimé du profil d'instance par l'action `RemoveRoleFromInstanceProfile` IAM. Vous pouvez utiliser l'action `AddRoleToInstanceProfile` IAM pour attacher un rôle au profil d'instance. Votre application devra attendre la prochaine actualisation planifiée pour accéder aux informations d'identification du rôle.

Pour forcer la modification, vous devez dissocier le profil d'instance, puis associer le profil d'instance, ou vous pouvez arrêter votre instance, puis la redémarrer.

Le document **iam/security-credentials/[role-name]** indique **"Code":"AssumeRoleUnauthorizedAccess"**

Amazon EC2 n'a pas l'autorisation d'endosser le rôle. L'autorisation d'endosser le rôle est contrôlée par la politique d'approbation attachée au rôle, comme dans l'exemple suivant. Utilisez l'API `UpdateAssumeRolePolicy` IAM pour mettre à jour la politique d'approbation.

```
{"Version": "2012-10-17","Statement": [{"Effect": "Allow","Principal": {"Service":
 ["ec2.amazonaws.com"]},"Action": ["sts:AssumeRole"]}]}
```

Votre application attendra jusqu'à la prochaine actualisation des informations d'identification planifiée automatiquement pour accéder aux informations d'identification du rôle.

Pour forcer la modification, vous devez dissocier le profil d'instance, puis associer le profil d'instance, ou vous pouvez arrêter votre instance, puis la redémarrer.

Résolution des problèmes liés à IAM et Amazon S3

Utilisez ces informations pour identifier et résoudre les problèmes courants que vous pouvez rencontrer lors de l'utilisation d'Amazon S3 et IAM.

Comment accorder un accès anonyme à un compartiment Amazon S3 ?

Vous utilisez une politique de compartiment Amazon S3 qui spécifie un caractère générique (*) dans l'élément `principal`, ce qui signifie que n'importe qui peut accéder au compartiment. Avec un accès anonyme, n'importe qui (y compris les utilisateurs sans identifiant Compte AWS) pourra accéder au bucket. Pour consulter un exemple de politique, veuillez consulter la rubrique Example Cases for Amazon S3 Bucket Policies (cas de politiques de compartiment Amazon S3) dans l'espace Amazon Simple Storage Service User Guide (guide de l'utilisateur du service simple de stockage Amazon).

Je suis connecté en tant qu'utilisateur Compte AWS root ; pourquoi ne puis-je pas accéder à un compartiment Amazon S3 sous mon compte ?

Dans certains cas, un utilisateur IAM peut disposer d'un accès complet à IAM et Amazon S3. Si l'utilisateur IAM attribue une politique de compartiment à un compartiment Amazon S3 sans le spécifier Utilisateur racine d'un compte AWS comme principal, l'utilisateur root se voit refuser l'accès à ce compartiment. Cependant, comme utilisateur racine, il est toujours possible d'accéder

au compartiment. Pour ce faire, modifiez la politique de compartiment pour autoriser l'accès à l'utilisateur racine à partir de la console Amazon S3 ou de la AWS CLI. Utilisez le principal suivant, en remplaçant *123456789012* par l'ID de l' Compte AWS.

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

Résolution des problèmes liés à la fédération SAML 2.0 avec AWS

Utilisez ces informations pour identifier et résoudre les problèmes courants que vous pouvez rencontrer lors de l'utilisation de SAML 2.0 et de la fédération avec IAM.

Rubriques

- Error: Your request included an invalid SAML response. To logout, click here.
- Erreur : RoleSessionName obligatoire dans AuthnResponse (service : AWSSecurityTokenService ; code d'état : 400 ; code d'erreur : InvalidIdentityToken)
- Erreur : Non autorisé à exécuter sts : AssumeRole WithSAML (service : AWSSecurityTokenService ; code d'état : 403 ; code d'erreur :) AccessDenied
- Erreur : l' RoleSessionName entrée AuthnResponse doit correspondre à [a-zA-Z_0-9+=, .@-] {2,64} (service : ; code d'état : 400 ; code d'erreur : AWSSecurityTokenService) InvalidIdentityToken
- Erreur : l'identité de la source doit correspondre à [a-zA-Z_0-9+=, .@-] {2,64} et ne pas commencer par "aws:" (service : ; code d'état : 400 ; code d'erreur : AWSSecurityTokenService) InvalidIdentityToken
- Erreur : signature de réponse non valide (service : AWSSecurityTokenService ; code d'état : 400 ; code d'erreur : InvalidIdentityToken)
- Erreur : Impossible d'assumer le rôle : l'émetteur n'est pas présent chez le fournisseur spécifié (service : AWSOpenIdDiscoveryService ; code d'état : 400 ; code d'erreur : AuthSamlInvalidSamlResponseException)
- Erreur : Impossible d'analyser les métadonnées.
- Erreur : Le fournisseur spécifié n'existe pas.
- Erreur : le nombre demandé DurationSeconds dépasse ce qui est MaxSessionDuration défini pour ce rôle.
- Erreur : la réponse ne contient pas l'audience requise.
- Comment afficher une réponse SAML dans votre navigateur à des fins de dépannage

Error: Your request included an invalid SAML response. To logout, click here.

Cette erreur peut se produire lorsque la réponse SAML du fournisseur d'identité n'inclut pas d'attribut avec le `Name` défini sur `https://aws.amazon.com/SAML/Attributes/Role`. L'attribut doit contenir un ou plusieurs éléments `AttributeValue`, chacun contenant une paire de chaînes séparées par une virgule :

- L'ARN d'un rôle auquel l'utilisateur peut être mappé
- L'ARN du fournisseur SAML

Pour plus d'informations, veuillez consulter Configurer les assertions SAML pour la réponse d'authentification. Pour afficher la réponse SAML dans votre navigateur, suivez les étapes répertoriées dans la section Comment afficher une réponse SAML dans votre navigateur à des fins de dépannage.

Erreur : RoleSessionName obligatoire dans AuthnResponse (service : AWSSecurityTokenService ; code d'état : 400 ; code d'erreur : InvalidIdentityToken)

Cette erreur peut se produire lorsque la réponse SAML du fournisseur d'identité n'inclut pas d'attribut avec le `Name` défini sur `https://aws.amazon.com/SAML/Attributes/RoleSessionName`. La valeur de l'attribut est un identifiant pour l'utilisateur, généralement un ID utilisateur ou une adresse e-mail.

Pour plus d'informations, veuillez consulter Configurer les assertions SAML pour la réponse d'authentification. Pour afficher la réponse SAML dans votre navigateur, suivez les étapes répertoriées dans la section Comment afficher une réponse SAML dans votre navigateur à des fins de dépannage.

Erreur : Non autorisé à exécuter sts : AssumeRole WithSAML (service : AWSSecurityTokenService ; code d'état : 403 ; code d'erreur :) AccessDenied

Cette erreur peut se produire si le rôle IAM spécifié dans la réponse SAML est mal orthographié ou n'existe pas. Assurez-vous d'utiliser le nom exact de votre rôle, car les noms de rôle sont sensibles à la casse. Corrigez le nom du rôle dans la configuration du fournisseur de service SAML.

Vous n'êtes autorisé à accéder que si votre politique d'approbation de rôle inclut l'action `sts:AssumeRoleWithSAML`. Si votre assertion SAML est configurée pour utiliser l'attribut `PrincipalTag`, votre politique d'approbation doit également inclure l'action `sts:TagSession`. Pour de plus amples informations sur les balises de session, veuillez consulter Transmission des balises de session AWS STS.

Cette erreur peut se produire si vous n'avez pas d'autorisations `sts:SetSourceIdentity` dans votre politique de confiance de rôle. Si votre assertion SAML est configurée pour utiliser l'attribut `SourceIdentity`, votre politique d'approbation doit également inclure l'action `sts:SetSourceIdentity`. Pour de plus amples informations sur l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

Cette erreur peut également se produire si les utilisateurs fédérés ne sont pas autorisés à endosser le rôle. Le rôle doit disposer d'une politique d'approbation qui spécifie l'ARN du fournisseur d'identité SAML IAM en tant que principal (`Principal`). Le rôle contient également des conditions qui contrôlent les utilisateurs qui peuvent endosser le rôle. Vérifiez que les utilisateurs satisfont les exigences des conditions.

Cette erreur peut également se produire si la réponse SAML n'inclut pas de `Subject` contenant un `NameID`.

Pour de plus amples informations, veuillez consulter Établir des autorisations dans AWS pour les utilisateurs fédérés et Configurer les assertions SAML pour la réponse d'authentification. Pour afficher la réponse SAML dans votre navigateur, suivez les étapes répertoriées dans la section Comment afficher une réponse SAML dans votre navigateur à des fins de dépannage.

Erreur : l' RoleSessionName entrée AuthnResponse doit correspondre à [a-zA-Z_0-9+=, .@-] {2,64} (service : ; code d'état : 400 ; code d'erreur : AWSSecurityTokenService) InvalidIdentityToken

Cette erreur peut se produire si la valeur d'attribut `RoleSessionName` est trop longue ou contient des caractères non valides. La longueur valide maximale est de 64 caractères.

Pour plus d'informations, veuillez consulter Configurer les assertions SAML pour la réponse d'authentification. Pour afficher la réponse SAML dans votre navigateur, suivez les étapes répertoriées dans la section Comment afficher une réponse SAML dans votre navigateur à des fins de dépannage.

Erreur : l'identité de la source doit correspondre à [a-zA-Z_0-9+=, .@-] {2,64} et ne pas commencer par **`"aws:"`** (service : ; code d'état : 400 ; code d'erreur : AWSSecurityTokenService) InvalidIdentityToken

Cette erreur peut se produire si la valeur d'attribut `sourceIdentity` est trop longue ou contient des caractères non valides. La longueur valide maximale est de 64 caractères. Pour de plus amples informations sur l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

Pour de plus amples informations sur la création d'assertions SAML, veuillez consulter Configurer les assertions SAML pour la réponse d'authentification. Pour afficher la réponse SAML dans votre navigateur, suivez les étapes répertoriées dans la section Comment afficher une réponse SAML dans votre navigateur à des fins de dépannage.

Erreur : signature de réponse non valide (service : AWSSecurityTokenService ; code d'état : 400 ; code d'erreur : InvalidIdentityToken)

Cette erreur peut se produire lorsque les métadonnées du fournisseur d'identité ne correspondent pas aux métadonnées du fournisseur d'identité IAM. Par exemple, le fichier de métadonnées du fournisseur de service d'identité peut avoir changé pour mettre à jour un certificat arrivé à expiration. Téléchargez le fichier de métadonnées SAML mis à jour depuis votre fournisseur de service d'identité. Mettez-le ensuite à jour dans l'entité du fournisseur AWS d'identité que vous définissez dans IAM à l'aide de la commande `aws iam update-saml-provider` CLI multiplateforme ou de l'`Update-IAMSAMLProvider` PowerShellapplet de commande.

Erreur : Impossible d'assumer le rôle : l'émetteur n'est pas présent chez le fournisseur spécifié (service : AWSOpenIdDiscoveryService ; code d'état : 400 ; code d'erreur : AuthSamlInvalidSamlResponseException)

Cette erreur peut se produire si l'émetteur de la réponse SAML ne correspond pas à l'émetteur déclaré dans le fichier de métadonnées de fédération. Le fichier de métadonnées a été chargé AWS lorsque vous avez créé le fournisseur d'identité dans IAM.

Erreur : Impossible d'analyser les métadonnées.

Cette erreur peut se produire si vous ne formatez pas correctement votre fichier de métadonnées.

Lorsque vous créez ou gérez un fournisseur d'identité SAML dans le AWS Management Console, vous devez récupérer le document de métadonnées SAML auprès de votre fournisseur d'identité.

Ce fichier de métadonnées inclut le nom de l'auteur, des informations d'expiration et des clés qui peuvent être utilisées pour valider les réponses d'authentification (assertions) SAML reçues du fournisseur d'identité (IdP). Le fichier de métadonnées doit être codé au format UTF-8 sans marque d'ordre d'octet (BOM). Pour supprimer la marque d'ordre d'octet (BOM), vous pouvez coder le fichier au format UTF-8 à l'aide d'un outil d'édition de texte, tel que Notepad++.

Le certificat x.509 inclus comme partie du document des métadonnées SAML doit utiliser une taille de clé au moins égale à 1 024 bits. De plus, le certificat x.509 doit également être exempt de toute extension répétée. Vous pouvez utiliser des extensions, mais elles ne peuvent apparaître qu'une seule fois dans le certificat. Si le certificat x.509 ne répond à aucune des conditions, la création de l'IdP échoue et renvoie une erreur « impossible d'analyser les métadonnées ».

Comme défini par la version 1.0 du profil d'interopérabilité des métadonnées SAML V2.0, IAM n'évalue ni ne prend aucune mesure concernant l'expiration du certificat X.509 du document de métadonnées.

Erreur : Le fournisseur spécifié n'existe pas.

Cette erreur peut se produire si le nom du fournisseur que vous spécifiez dans l'assertion SAML ne correspond pas au nom du fournisseur configuré dans IAM. Pour plus d’informations sur l'affichage du nom du fournisseur, veuillez consulter Création d'un fournisseur d'identité SAML dans IAM.

Erreur : le nombre demandé DurationSeconds dépasse ce qui est MaxSessionDuration défini pour ce rôle.

Cette erreur peut se produire si vous assumez un rôle depuis l'API AWS CLI or.

Lorsque vous utilisez la CLI assume-role-with-saml AssumeRole ou les opérations d'API WithSAML pour assumer un rôle, vous pouvez spécifier une valeur pour le paramètre. `DurationSeconds` Vous pouvez spécifier une valeur comprise entre 900 secondes (15 minutes) et la durée maximale de la session définie pour le rôle. Si vous spécifiez une valeur supérieure à ce paramètre, l'opération échoue. Par exemple, si vous spécifiez une durée de session de 12 heures, mais que votre administrateur a défini la durée de session maximale à 6 heures, votre opération échoue. Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Affichage du paramètre de durée de session maximale pour un rôle.

Erreur : la réponse ne contient pas l'audience requise.

Cette erreur peut se produire s'il existe une incompatibilité entre l'URL d'audience et le fournisseur d'identité dans la configuration SAML. Assurez-vous que l'identifiant de la partie dépendante de votre fournisseur d'identité (IdP) correspond exactement à l'URL d'audience (ID d'entité) fournie dans la configuration SAML.

Comment afficher une réponse SAML dans votre navigateur à des fins de dépannage

Les procédures suivantes décrivent l'affichage de la réponse SAML de votre fournisseur de services dans votre navigateur lors du dépannage d'un problème lié à SAML 2.0.

Pour tous les navigateurs, accédez à la page où vous pouvez reproduire le problème. Ensuite, suivez les étapes correspondant au navigateur approprié :

Rubriques

- Google Chrome
- Mozilla Firefox
- Apple Safari
- Que faire de la réponse SAML encodée au format Base64

Google Chrome

Pour afficher une réponse SAML dans Chrome

Ces étapes ont été testées à l'aide de la version 106.0.5249.103 (build officielle) (arm64) de Google Chrome. Si vous utilisez une autre version, il vous faudra peut-être modifier les étapes en conséquence.

1. Appuyez sur F12 pour démarrer la console Outils pour développeurs.
2. Sélectionnez l'onglet Réseau, puis sélectionnez Conserver le journal en haut à gauche de la fenêtre Outils pour développeurs.
3. Reproduisez le problème.
4. (Facultatif) Si la colonne Méthode n'est pas visible dans le volet du journal Outils pour développeurs Réseau, cliquez avec le bouton droit sur l'étiquette d'une colonne et choisissez Méthode pour ajouter la colonne.
5. Recherchez une Publication SAML dans le volet du journal Outils pour développeurs Réseau. Sélectionnez cette ligne, puis affichez l'onglet Charge utile en haut. Recherchez l'élément SAMLResponse contenant la demande encodée. La valeur associée est la réponse encodée au format Base64.

Mozilla Firefox

Pour afficher une réponse SAML dans Firefox

Cette procédure a été testée avec la version 105.0.3 (64 bits) de Mozilla Firefox. Si vous utilisez une autre version, il vous faudra peut-être modifier les étapes en conséquence.

1. Appuyez sur F12 pour démarrer la console Outils Web Developer.
2. Sélectionnez l'onglet Réseau.
3. Dans le coin supérieur droit de la fenêtre Outils Web Developer, cliquez sur options (petite icône d'engrenage). Sélectionnez Conserver les journaux.
4. Reproduisez le problème.
5. (Facultatif) Si la colonne Méthode n'est pas visible dans le volet du journal Outils Web Developer Réseau, cliquez avec le bouton droit sur l'étiquette d'une colonne et choisissez Méthode pour ajouter la colonne.

6. Recherchez un POST SAML dans le tableau. Sélectionnez cette ligne, puis affichez l'onglet Requête et trouvez l'élément SamlResponse. La valeur associée est la réponse encodée au format Base64.

Apple Safari

Pour afficher une réponse SAML dans Safari

Ces étapes ont été testées à l'aide de la version 16.0 (17614.1.25.9.10, 17614) d'Apple Safari. Si vous utilisez une autre version, il vous faudra peut-être modifier les étapes en conséquence.

1. Activez Web Inspector dans Safari. Ouvrez la fenêtre Préférences, sélectionnez l'onglet Avancées, puis sélectionnez Afficher le menu Développement dans la barre de menus.
2. Vous pouvez maintenant ouvrir Web Inspector. Choisissez Développement dans la barre de menu, puis sélectionnez Afficher Web Inspector.
3. Sélectionnez l'onglet Réseau.
4. Dans l'angle supérieur gauche de la fenêtre Web Inspector, choisissez les options (l'icône représentant un petit cercle contenant trois lignes horizontales). Sélectionnez Conserver le journal.
5. (Facultatif) Si la colonne Méthode n'est pas visible dans le volet du journal Web Inspector Réseau, cliquez avec le bouton droit sur l'étiquette d'une colonne et choisissez Méthode pour ajouter la colonne.
6. Reproduisez le problème.
7. Recherchez un POST SAML dans le tableau. Sélectionnez cette ligne, puis affichez l'onglet En-têtes.
8. Recherchez l'élément SAMLResponse contenant la demande encodée. Faites défiler la liste pour localiser l'élément `Request Data` nommé `SAMLResponse`. La valeur associée est la réponse encodée au format Base64.

Que faire de la réponse SAML encodée au format Base64

Une fois que vous avez trouvé l'élément contenant la réponse SAML encodée au format Base64 dans votre navigateur, copiez-le et utilisez votre outil de décodage Base64 favori pour extraire la réponse avec balise XML.

> **Conseil de sécurité**
>
> Dans la mesure où les données de réponse SAML que vous affichez peuvent être des données de sécurité sensibles, il est recommandé de ne pas utiliser un décodeur Base64 en ligne. Préférez un outil installé sur un ordinateur local qui n'achemine pas les données SAML via le réseau.

Option intégrée pour les systèmes Windows (PowerShell) :

```
PS C:
\> [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String("base64encodedtext
```

Option intégrée pour les systèmes MacOS et Linux :

```
$ echo "base64encodedtext" | base64 --decode
```

Vérifiez les valeurs du fichier décodé

Vérifiez les valeurs du fichier de réponse SAML décodé.

- Vérifiez que la valeur de l'attribut SAML:NameID correspond au nom d'utilisateur de l'utilisateur authentifié.
- Vérifiez la valeur de https://aws.amazon.com/SAML/Attributes/Role. L'ARN et le fournisseur SAML distinguent les majuscules et minuscules, et l'ARN doit correspondre à la ressource de votre compte.
- Vérifiez la valeur de https://aws.amazon.com/SAML/Attributes/ RoleSession Name. La valeur doit correspondre à la valeur de la règle de réclamation.
- Si vous configurez la valeur d'attribut pour une adresse e-mail ou un nom de compte, assurez-vous que les valeurs sont correctes. Les valeurs doivent correspondre à l'adresse e-mail ou au nom de compte de l'utilisateur authentifié.

Vérifiez les erreurs et confirmez la configuration

Vérifiez si les valeurs contiennent des erreurs et confirmez que les configurations suivantes sont correctes.

- Les règles de réclamation répondent aux éléments requis et tous les ARN sont corrects. Pour plus d'informations, consultez Configurez votre IdP SAML 2.0 en vous fiant à la confiance des parties et en ajoutant des réclamations.
- Vous avez chargé le dernier fichier de métadonnées de votre IdP AWS dans votre fournisseur SAML. Pour plus d'informations, consultez Permettre aux utilisateurs fédérés SAML 2.0 d'accéder au AWS Management Console.
- Vous avez correctement configuré la politique de confiance du rôle IAM. Pour plus d'informations, voir Modification d'un rôle.

Informations de référence pour AWS Identity and Access Management

Utilisez les rubriques de cette section pour trouver des documents de référence détaillés sur différents aspects d'IAM et AWS STS.

Rubriques

- Amazon Resource Names (ARN)
- Identifiants IAM
- IAM et quotas AWS STS
- Points de terminaison de VPC d’Interface
- AWS services qui fonctionnent avec IAM
- Signature des demandes AWS d'API
- Référence de politique JSON IAM

Amazon Resource Names (ARN)

Les Amazon Resource Names (ARN) identifient les AWS ressources de manière unique. Nous avons besoin d'un ARN lorsque vous devez spécifier une ressource sans ambiguïté dans l'ensemble, par exemple dans les politiques IAM AWS, les balises Amazon Relational Database Service (Amazon RDS) et les appels d'API.

Format ARN

Voici les formats généraux des ARN. Les formats spécifiques dépendent de la ressource. Pour utiliser un ARN, remplacez le texte en *`italique`* par les informations spécifiques à la ressource. Sachez que les ARN de certaines ressources omettent la région, l'ID du compte ou la région et l'ID du compte.

```
arn:partition:service:region:account-id:resource-id
arn:partition:service:region:account-id:resource-type/resource-id
arn:partition:service:region:account-id:resource-type:resource-id
```

partition

Partition dans laquelle se trouve la ressource. Une partition est un groupe de AWS régions. Chaque AWS compte est limité à une partition.

Les partitions prises en charge sont les suivantes :

- `aws`- AWS Régions
- `aws-cn` - Régions chinoises
- `aws-us-gov` - AWS GovCloud (US) Régions

service

L'espace de noms du service qui identifie le AWS produit.

region

Code de région. Par exemple, indiquez `us-east-2` pour la région USA Est (Ohio). Pour obtenir la liste des codes de région, consultez Points de terminaison régionaux dans Références générales AWS.

account-id

L'ID du AWS compte propriétaire de la ressource, sans les traits d'union. Par exemple, `123456789012`.

resource-type

Type de ressource. Par exemple, `vpc` pour un cloud privé virtuel (VPC)

resource-id

Identificateur de la ressource. Il s'agit du nom de la ressource, de l'ID de la ressource ou d'un chemin de ressource. Certains identificateurs de ressources incluent une ressource parent (sub-resource-type/parent-resource/sub-resource) ou un qualificatif tel qu'une version (resource-type:resource-name:qualifier).

Exemples

Utilisateur IAM

arn:aws:iam::*123456789012*:user/*johndoe*

Rubrique SNS

arn:aws:sns : us-east-1 : 123456789012 : example-sns-topic-name

VPC

arn:aws:ec2:*us-east-1*:*123456789012*:vpc/*vpc-0e9801d129EXAMPLE*

Rechercher le format d'ARN pour une ressource

Le format exact d'un ARN dépend du service et du type de ressource. Certains ARN de ressources peuvent inclure un chemin, une variable ou un caractère générique. Pour rechercher le format ARN d'une AWS ressource spécifique, ouvrez la référence d'autorisation du service, ouvrez la page du service et accédez au tableau des types de ressources.

Chemins d'accès dans les ARN

Certains ARN de ressource peuvent inclure un chemin. Par exemple, dans Amazon S3, l'identificateur de ressource est un nom d'objet qui peut inclure des barres obliques (/) pour former un chemin d'accès. De même, les noms d'utilisateur et les noms de groupe IAM peuvent inclure des chemins d'accès. Seuls les caractères alphanumériques et les caractères suivants sont autorisés dans les chemins IAM : barre oblique (/), plus (+), égal (=), virgule (,), point (.), arobase (@), trait de soulignement (_), et trait d'union (-).

Utilisation de caractères génériques dans les chemins

Les chemins peuvent inclure un caractère générique, à savoir un astérisque (*). Par exemple, si vous écrivez une politique IAM, vous pouvez spécifier tous les utilisateurs IAM ayant le chemin d'accès `product_1234` à l'aide d'un caractère générique comme suit :

```
arn:aws:iam::123456789012:user/Development/product_1234/*
```

De même, vous pouvez spécifier `user/*` pour indiquer tous les utilisateurs ou `group/*` pour indiquer tous les groupes, comme dans les exemples suivants :

```
"Resource":"arn:aws:iam::123456789012:user/*"
"Resource":"arn:aws:iam::123456789012:group/*"
```

L'exemple suivant illustre les ARN d'un compartiment Amazon S3 dans lequel le nom de la ressource inclut un chemin d'accès :

```
arn:aws:s3:::my_corporate_bucket/*
arn:aws:s3:::my_corporate_bucket/Development/*
```

Utilisation incorrecte des caractères génériques

Vous ne pouvez pas utiliser de caractère générique dans la partie de l'ARN qui spécifie le type de ressource, comme le terme `user` d'un ARN IAM. Par exemple, ce qui suit n'est pas autorisé.

```
arn:aws:iam::123456789012:u*   <== not allowed
```

Identifiants IAM

IAM utilise quelques identifiants différents pour les utilisateurs, groupes d'utilisateurs, rôles, politiques et certificats de serveur. Cette section décrit les identifiants et quand vous utilisez chacun d'eux.

Rubriques

- Noms conviviaux et chemins
- ARN IAM
- Identifiants uniques

Noms conviviaux et chemins

Lorsque vous créez un utilisateur, un rôle, un groupe d'utilisateurs ou une politique, ou lorsque vous téléchargez un certificat de serveur, vous lui attribuez un nom convivial. Les exemples incluent Bob, TestApp 1, Developers ManageCredentialsPermissions, ou ProdServerCert.

Si vous utilisez l'API IAM ou AWS Command Line Interface (AWS CLI) pour créer des ressources IAM, vous pouvez ajouter un chemin facultatif. Vous pouvez utiliser un seul chemin ou imbriquer plusieurs chemins en une structure de dossiers. Par exemple, vous pouvez utiliser le chemin imbriqué `/division_abc/subdivision_xyz/product_1234/engineering/` pour correspondre à l'organigramme de votre société. Vous pouvez ensuite créer une politique pour autoriser tous les utilisateur dans ce chemin à accéder à l'API de simulateur de politique. Pour afficher cette politique, consultez IAM : accès à l'API du simulateur de politique en fonction du chemin d'utilisateur. Pour plus d'informations sur la façon dont un nom convivial peut être spécifié, consultez la documentation de l'API utilisateur. Pour trouver des exemples de la manière dont vous pouvez utiliser des chemins, consultez ARN IAM.

Lorsque vous créez des ressources, vous pouvez spécifier un chemin pour les utilisateurs, les groupes d'utilisateurs et les rôles, ainsi que des politiques gérées par le client. AWS CloudFormation

Si vous avez un utilisateur et un groupe d'utilisateurs sur le même chemin, IAM ne place pas automatiquement l'utilisateur dans ce groupe d'utilisateurs. Par exemple, vous pouvez créer un groupe d'utilisateurs Développeurs et spécifier le chemin d'accès comme `/division_abc/subdivision_xyz/product_1234/engineering/`. Si vous créez un utilisateur nommé Bob et lui ajoutez le même chemin, cela ne place pas Bob automatiquement dans le groupe d'utilisateurs Développeurs. IAM n'impose aucune limite entre utilisateurs ou groupes d'utilisateurs en fonction de leurs chemins. Les utilisateurs ayant des chemins différents peuvent utiliser les mêmes ressources en supposant qu'ils aient autorisation d'accéder à ces ressources. Le nombre et la taille des ressources IAM d'un AWS compte sont limités. Pour plus d'informations, consultez IAM et quotas AWS STS.

ARN IAM

Les plupart des ressources ont un nom convivial, par exemple, un utilisateur appelé `Bob` ou un groupe d'utilisateurs appelé `Developers`. Toutefois, le langage de politique d'autorisation nécessite que vous indiquiez la ou les ressources au format Amazon Resource Name (ARN) suivant.

```
arn:partition:service:region:account:resource
```

Où :

- `partition` identifie la partition pour la ressource. Pour les Régions AWS standards, la partition est `aws`. Si vous avez des ressources dans d'autres partitions, la partition est `aws-`*`partitionname`*. Par exemple, la partition des ressources de la région Chine (Beijing) est `aws-cn`. Vous ne pouvez pas déléguer l'accès entre comptes dans différentes partitions.
- `service`identifie le AWS produit. Les ressources IAM utilisent toujours `iam`.
- `region` identifie la région de la ressource. Pour les ressources IAM, elle est toujours laissée vide.
- `account`spécifie l' Compte AWS ID sans tiret.
- `resource` identifie la ressources spécifique par nom.

Vous pouvez spécifier IAM et AWS STS ARN à l'aide de la syntaxe suivante. La partie de la région de l'ARN est vide, car les ressources IAM sont globales.

Syntaxe :

```
arn:aws:iam::account:root
arn:aws:iam::account:user/user-name-with-path
```

```
arn:aws:iam::account:group/group-name-with-path
arn:aws:iam::account:role/role-name-with-path
arn:aws:iam::account:policy/policy-name-with-path
arn:aws:iam::account:instance-profile/instance-profile-name-with-path
arn:aws:sts::account:federated-user/user-name
arn:aws:sts::account:assumed-role/role-name/role-session-name
arn:aws:sts::account:self
arn:aws:iam::account:mfa/virtual-device-name-with-path
arn:aws:iam::account:u2f/u2f-token-id
arn:aws:iam::account:server-certificate/certificate-name-with-path
arn:aws:iam::account:saml-provider/provider-name
arn:aws:iam::account:oidc-provider/provider-name
```

Bon nombre des exemples suivants incluent des chemins dans la partie des ressources de l'ARN. Les chemins ne peuvent pas être créés ou manipulés dans l'interface AWS Management Console. Pour utiliser des chemins, vous devez utiliser la ressource à l'aide de l' AWS API AWS CLI, du ou des outils pour Windows PowerShell.

Exemples :

```
arn:aws:iam::123456789012:root
arn:aws:iam::123456789012:user/JohnDoe
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/JaneDoe
arn:aws:iam::123456789012:group/Developers
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
arn:aws:iam::123456789012:role/S3Access
arn:aws:iam::123456789012:role/application_abc/component_xyz/RDSAccess
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
arn:aws:iam::123456789012:role/service-role/QuickSightAction
arn:aws:iam::123456789012:policy/UsersManageOwnCredentials
arn:aws:iam::123456789012:policy/division_abc/subdivision_xyz/UsersManageOwnCredentials
arn:aws:iam::123456789012:instance-profile/Webserver
arn:aws:sts::123456789012:federated-user/JohnDoe
arn:aws:sts::123456789012:assumed-role/Accounting-Role/JaneDoe
arn:aws:sts::123456789012:self
arn:aws:iam::123456789012:mfa/JaneDoeMFA
arn:aws:iam::123456789012:u2f/user/JohnDoe/default (U2F security key)
arn:aws:iam::123456789012:server-certificate/ProdServerCert
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
```

```
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
arn:aws:iam::123456789012:oidc-provider/server.example.org
```

Les exemples suivants fournissent plus de détails pour vous aider à comprendre le format ARN pour différents types d'IAM et de AWS STS ressources.

- Un utilisateur IAM dans le compte :

> **Note**
>
> Chaque nom d'utilisateur IAM est unique. Le nom d'utilisateur est sensible à la casse pour l'utilisateur, par exemple lors du processus de connexion, mais il l'est également lorsque vous l'utilisez dans une politique ou dans le cadre d'un ARN.

```
arn:aws:iam::123456789012:user/JohnDoe
```

- Un autre utilisateur avec un chemin reflétant un organigramme :

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/JaneDoe
```

- Pour un groupe d'utilisateurs IAM :

```
arn:aws:iam::123456789012:group/Developers
```

- Un groupe d'utilisateurs IAM avec un chemin :

```
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
```

- Un rôle IAM :

```
arn:aws:iam::123456789012:role/S3Access
```

- Un rôle lié à un service :

```
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
```

- Un rôle de service :

```
arn:aws:iam::123456789012:role/service-role/QuickSightAction
```

- Une politique gérée :

```
arn:aws:iam::123456789012:policy/ManageCredentialsPermissions
```

- Un profil d'instance qui peut être associé à une instance Amazon EC2 :

```
arn:aws:iam::123456789012:instance-profile/Webserver
```

- Un utilisateur fédéré identifié dans IAM sous le nom de « Paulo » :

```
arn:aws:sts::123456789012:federated-user/Paulo
```

- La session active d'une personne endossant le rôle de « Comptable », avec le nom de session de rôle « Mary » :

```
arn:aws:sts::123456789012:assumed-role/Accounting-Role/Mary
```

- Représente la propre session de l'appelant lorsqu'elle est utilisée comme ressource dans un appel d'API, tel que l' AWS STS SetContextAPI, qui fonctionne sur la session d'appel :

```
arn:aws:sts::123456789012:self
```

- L'appareil MFA (Multi-Factor Authentication) attribué à l'utilisateur nommé Jorge :

```
arn:aws:iam::123456789012:mfa/Jorge
```

- Un certificat de serveur :

```
arn:aws:iam::123456789012:server-certificate/ProdServerCert
```

- Un certificat de serveur avec un chemin qui reflète un organigramme :

```
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
```

- Fournisseurs d'identité (SAML et OIDC) :

```
arn:aws:iam::123456789012:saml-provider/ADFSProvider
```

```
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
arn:aws:iam::123456789012:oidc-provider/server.example.org
```

- Un fournisseur d'identité OIDC avec un chemin qui reflète l'URL d'un fournisseur d'identité OIDC d'Amazon EKS :

```
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
```

Autre ARN important : celui de l'utilisateur racine. Bien qu'il ne s'agisse pas d'une ressource IAM, vous devez connaître le format de cet ARN. Il est souvent utilisé dans l'élément `Principal` d'une stratégie.

- Compte AWS Affiche les informations suivantes :

```
arn:aws:iam::123456789012:root
```

L'exemple suivant montre une politique que vous pouvez affecter à Richard afin de lui permettre de gérer ses clés d'accès. Notez que la ressource est l'utilisateur IAM Richard.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ManageRichardAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/division_abc/subdivision_xyz/Richard"
        },
        {
            "Sid": "ListForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "*"
        }
    ]
```

```
}
```

> **Note**
>
> Lorsque vous utilisez des ARN pour identifier des ressources dans une politique IAM, vous pouvez inclure des variables de politique. Les variables de politique peuvent inclure des espaces réservés pour les informations d'exécution (notamment le nom de l'utilisateur) comme faisant partie de l'ARN. Pour plus d'informations, veuillez consulter Éléments des politiques IAM : variables et balises.

Utilisation de caractères génériques et de chemins dans les ARN

Vous pouvez utiliser des caractères génériques dans la partie *ressource* de l'ARN pour spécifier plusieurs utilisateurs, groupes d'utilisateurs ou politiques. Par exemple, pour spécifier tous les utilisateurs travaillant sur le produit_1234, vous utilisez :

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/*
```

Si vous avez des utilisateurs dont le nom commence par la chaîne `app_`, vous pouvez faire référence à tous ces utilisateurs avec l'ARN suivant.

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/app_*
```

Pour spécifier tous les utilisateurs, groupes d'utilisateurs ou politiques de votre Compte AWS compte, utilisez un caractère générique après le `user/group/`, ou une `policy/` partie de l'ARN, respectivement.

```
arn:aws:iam::123456789012:user/*
arn:aws:iam::123456789012:group/*
arn:aws:iam::123456789012:policy/*
```

Si vous spécifiez l'ARN suivant pour un utilisateur `arn:aws:iam::111122223333:user/*`, il correspond aux deux exemples suivants.

```
arn:aws:iam::111122223333:user/JohnDoe
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

Mais, si vous spécifiez l'ARN suivant pour un utilisateur `arn:aws:iam::111122223333:user/division_abc*`, il correspond au second exemple, mais pas au premier.

```
arn:aws:iam::111122223333:user/JohnDoe
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

N'utilisez pas un caractère générique dans la partie `user/`, `group/` ou `policy/` de l'ARN. Par exemple, IAM n'autorise pas ce qui suit :

```
arn:aws:iam::123456789012:u*
```

Example Exemple d'utilisation des chemins et des ARN d'un groupe d'utilisateurs basé sur un projet

Les chemins ne peuvent pas être créés ou manipulés dans l'interface AWS Management Console. Pour utiliser des chemins, vous devez utiliser la ressource à l'aide de l' AWS API AWS CLI, du ou des outils pour Windows PowerShell.

Dans cet exemple, Jules dans le groupe d'utilisateurs Marketing_Admin crée un groupe d'utilisateurs basé sur un projet dans le chemin /marketing/. Il attribue des utilisateurs de différentes parties de l'entreprise au groupe d'utilisateurs. Cet exemple illustre que le chemin d'un utilisateur n'est pas lié aux groupes d'utilisateurs dont fait partie l'utilisateur.

Le groupe de marketing a un nouveau produit à lancer, Jules crée un groupe d'utilisateurs sur le chemin /marketing/ appelé Widget_Launch. Jules affecte ensuite la politique suivante au groupe d'utilisateurs qui accorde à celui-ci l'accès aux objets situés dans la partie du `example_bucket` désigné pour ce lancement spécifique.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::example_bucket/marketing/newproductlaunch/widget/*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket*",
      "Resource": "arn:aws:s3:::example_bucket",
      "Condition": {"StringLike": {"s3:prefix": "marketing/newproductlaunch/widget/*"}}
    }
```

```
  ]
}
```

Jules affecte ensuite les utilisateurs qui travaillent sur ce lancement au groupe d'utilisateurs. Cela inclut Patricia et Eli du chemin /marketing/. Cela inclut également Chris et Chloe du chemin /sales/ et Alice et Jim du chemin /legal/.

Identifiants uniques

Lorsqu'IAM crée un utilisateur, un groupe d'utilisateurs, un rôle, une stratégie, un profil d'instance ou un certificat de serveur, il accorde à chaque ressource un ID unique. L'ID se présente comme suit :

`AIDAJQABLZS4A3QDU576Q`

En règle générale, vous avez recours à des noms conviviaux et à des ARN lorsque vous utilisez des ressources IAM. De la sorte, vous n'avez pas besoin de connaître l'ID unique d'une ressource spécifique. En revanche, l'ID unique peut parfois être utile quand il n'est pratique d'utiliser des noms conviviaux.

Un exemple réutilise des noms conviviaux dans votre Compte AWS. Dans votre compte, un nom convivial pour un utilisateur, un groupe d'utilisateurs ou une stratégie doivent être uniques. Par exemple, vous pouvez créer un utilisateur IAM appelé `John`. Votre entreprise utilise Amazon S3 et dispose d'un compartiment avec des dossiers pour chaque employé. L'utilisateur IAM `John` est membre d'un groupe d'utilisateurs IAM appelé `User-S3-Access` avec des autorisations qui permettent aux utilisateurs d'accéder uniquement à leurs propres dossiers dans le compartiment. Pour obtenir un exemple de création d'une stratégie basée sur l'identité qui autorise des utilisateurs IAM à accéder à leur propre objet de compartiment dans S3 à l'aide du nom convivial des utilisateurs, consultez Amazon S3 : autorise les utilisateurs IAM à accéder à leur répertoire de base S3, par programmation et dans la console.

Supposons que l'employé appelé John quitte votre entreprise et que vous supprimez l'utilisateur IAM correspondant appelé `John`. Mais plus tard, un autre employé appelé John commence à travailler pour votre entreprise et vous créez un utilisateur IAM appelé `John`. Vous ajoutez le nouvel utilisateur IAM appelé `John` au groupe d'utilisateurs IAM `User-S3-Access` existant. Si la stratégie associée au groupe d'utilisateurs spécifie le nom d''utilisateur IAM convivial `John`, la stratégie permet au nouveau John d'accéder aux informations laissées par l'ancien John.

En règle générale, nous vous recommandons de spécifier l'ARN de la ressource dans vos politiques au lieu de son ID unique. En revanche, chaque utilisateur IAM dispose d'un ID unique, même si vous

créez un utilisateur IAM qui réutilise un nom convivial que vous avez supprimé avant. Dans l'exemple, l'ancien utilisateur IAM `John` et le nouvel utilisateur IAM `John` ont des ID uniques distincts. Vous pouvez créer des politiques basées sur les ressources qui accordent l'accès par ID unique et pas seulement par nom d'utilisateur. Vous réduisez ainsi le risque d'accorder par inadvertance l'accès à des informations dont un employé ne devrait pas disposer.

L'exemple suivant montre comment spécifier des ID uniques dans l'élément `Principal` principal d'une stratégie basée sur les ressources.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::111122223333:role/role-name",
    "AIDACKCEVSQ6C2EXAMPLE",
    "AROADBQP57FF2AEXAMPLE"
  }
```

L'exemple suivant montre comment spécifier des ID uniques dans l'élément `Condition` principal d'une stratégie en utilisant une clé de condition globale `aws:userid`.

```
"Condition": {
    "StringLike": {
      "aws:userId": [
        "AIDACKCEVSQ6C2EXAMPLE",
        "AROADBQP57FF2AEXAMPLE:role-session-name",
        "AROA1234567890EXAMPLE:*",
        "111122223333"
      ]
    }
  }
```

Un autre exemple dans lequel les ID d'utilisateur puevent être utiles c'est lorsque vous maintenez votre propre base de données (ou autre moyen de stockage) d'un utilisateur IAM ou d'informations sur les rôles. L'ID unique peut fournir un identifiant unique pour chaque utilisateur IAM que vous créez. Cela reste vrai lorsque vous avez des utilisateurs IAM ou des rôles qui réutilisent un nom, comme dans l'exemple précédent.

Présentation des préfixes d'ID uniques

IAM utilise les préfixes suivants pour indiquer le type de ressource auquel chaque ID unique s'applique. Les préfixes peuvent varier en fonction de leur date de création.

Préfixe	Type de ressource
ABIA	AWS STS jeton de support
ACCA	Informations d'identification spécifiques au contexte
AGPA	Groupe d'utilisateurs
AIDA	Utilisateur IAM
AIPA	Profil d'instance Amazon EC2
AKIA	Clé d'accès
ANPA	Politique gérée
ANVA	La version d'une politique gérée
APKA	Clé publique
AROA	Rôle
ASCA	Certificat
ASIA	Les ID de clé d'accès temporaire (AWS STS) utilisent ce préfixe, mais ne sont uniques qu'en combinaison avec la clé d'accès secrète et le jeton de session.

Obtention de l'identifiant unique

L'ID unique d'une ressource IAM n'est pas disponible dans la console IAM. Pour obtenir l'identifiant unique, vous pouvez utiliser les AWS CLI commandes ou les appels d'API IAM suivants.

AWS CLI:

- get-caller-identity
- get-group
- get-role

- get-user
- get-policy
- get-instance-profile
- get-server-certificate

API IAM

- GetCallerIdentity
- GetGroup
- GetRole
- GetUser
- GetPolicy
- GetInstanceProfile
- GetServerCertificate

IAM et quotas AWS STS

AWS Identity and Access Management (IAM) et AWS Security Token Service (STS) ont des quotas qui limitent la taille des objets. Cela affecte la façon dont vous nommez un objet, le nombre d'objets que vous pouvez créer et le nombre de caractères que vous pouvez utiliser lorsque vous transmettez un objet.

> **Note**
>
> Pour obtenir des informations au niveau du compte sur l'utilisation et les quotas d'IAM, utilisez l'opération d'GetAccountSummaryAPI ou la commande. get-account-summary AWS CLI

Exigences relatives aux noms IAM

Les noms IAM ont les exigences et restrictions suivantes :

- Les documents de politique 000A peuvent contenir uniquement les caractères Unicode suivants : tabulation horizontale (U+0009), saut de ligne (U+000A), retour chariot (U+000D) et caractères de la plage U+0020 à U+00FF.

- Les noms d’utilisateurs, de groupes, de rôles, de politiques, de profils d’instance et de certificats de serveur et de chemins doivent être alphanumériques, comprennant les caractères communs suivants : plus (+), égal (=), virgule (,), point (.), arobase (@), trait de soulignement (_) et trait d’union (-). Les noms de chemins doivent commencer et finir par une barre oblique (/).
- Les noms des utilisateurs, des groupes, des rôles et des profils d'instance doivent être uniques au sein du compte. La casse majuscules-minuscules ne compte pas. Par exemple, vous ne pouvez pas créer en même temps des groupes nommés **`ADMINS`** et **`admins`**.
- La valeur d'ID externe qu'un tiers utilise pour endosser un rôle doit avoir au minimum 2 caractères et au maximum 1 224 caractères. La valeur doit être alphanumérique sans espaces. Elle peut également inclure les symboles suivants : signe plus (+), signe égal (=), virgule (,), point (.), arobase (@), deux points (:), barre oblique (/) et tiret (-). Pour plus d'informations sur l'ID externe, consultez Comment utiliser un identifiant externe lorsque vous accordez l'accès à vos AWS ressources à un tiers.
- Les noms de politique pour les politiques en ligne doivent être uniques pour l'utilisateur, le groupe ou le rôle auquel ils sont intégrés. Les noms peuvent contenir tous les caractères latins de base (ASCII), à l'exception des caractères réservés suivants : barre oblique inversée (\), barre oblique (/), astérisque (*), point d'interrogation (?) et espace. Ces caractères sont réservés conformément à la norme RFC 3986, section 2.2.
- Les mots de passe des utilisateurs (profils de connexion) peuvent contenir tous les caractères latins (ASCII) de base.
- Compte AWS Les alias d'identification doivent être uniques pour tous les AWS produits et doivent être alphanumériques conformément aux conventions de dénomination DNS. Un alias doit être en minuscules, ne doit pas commencer ou se terminer par un trait d'union, ne peut pas contenir deux traits d'union consécutifs et ne peut pas être un numéro à 12 chiffres.

Pour obtenir une liste des caractères latins (ASCII) de base, accédez à la Library of Congress Basic Latin (ASCII) Code Table.

Quotas d'objet IAM

Les quotas, également appelés limites dans AWS, sont les valeurs maximales pour les ressources, les actions et les éléments de votre Compte AWS. Utilisez Service Quotas pour gérer vos quotas IAM.

Pour obtenir la liste des points de terminaison et des quotas de service IAM, consultez Points de terminaison et quotas de service AWS Identity and Access Management (français non garanti) dans Références générales AWS.

Pour demander une augmentation de quota

1. Suivez la procédure de connexion correspondant à votre type d'utilisateur, comme décrit dans la rubrique Connexion à AWS (français non garanti) du Guide de l'utilisateur Connexion àAWS pour vous connecter à AWS Management Console.
2. Ouvrez la console Service Quotas.
3. Dans le panneau de navigation, choisissez Services AWS .
4. Dans la barre de navigation, sélectionnez la région US East (N. Virginia). Ensuite, recherchez **IAM**.
5. Sélectionnez AWS Identity and Access Management (IAM), choisissez un quota et suivez les instructions pour demander une augmentation de quota.

Pour de plus amples informations, veuillez consulter Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

Pour voir un exemple de demande d'augmentation de quota IAM à l'aide de la console Service Quotas, regardez la vidéo suivante.

Demandez une augmentation de quota IAM à l'aide de la console Service Quotas.

Vous pouvez demander une augmentation des quotas par défaut pour les quotas ajustables IAM. Les demandes jusqu'à maximum quota sont automatiquement approuvées et terminées en quelques minutes.

Le tableau suivant répertorie les ressources pour lesquelles les augmentations de quotas peuvent être automatiquement approuvées.

Quotas ajustables pour les ressources IAM

Ressource	Quota par défaut	Quota maximal
Politiques gérées par le client par compte	1 500	5000
Groupes par compte	300	500

Ressource	Quota par défaut	Quota maximal
Profils d'instances par compte	1 000	5000
Politiques gérées par rôle	10	20
Politiques gérées par utilisateur	10	20
Longueur de la politique d'approbation du rôle	2048 caractères	4096 caractères
Rôles par compte	1 000	5000
Certificats de serveur par compte	20	1 000

Quotas de l'IAM Access Analyzer

Pour obtenir la liste des points de terminaison et des quotas de service de l'analyseur d'accès IAM, consultez Points de terminaison et quotas de l'analyseur d'accès IAM (français non garanti) dans Références générales AWS.

Quotas des rôles Anywhere IAM

Pour obtenir la liste des points de terminaison et des quotas de service des rôles Anywhere IAM, consultez Points de terminaison et quotas de service des rôles Anywhere AWS Identity and Access Management (français non garanti) dans Références générales AWS.

Limites des caractères d'IAM et de STS

Le nombre maximal de caractères et les limites de taille pour IAM et AWS STS sont les suivants. Vous ne pouvez pas demander une augmentation pour les limites suivantes.

Description	Limite
Alias pour un Compte AWS identifiant	3–63 caractères

Description	Limite
Pour les politiques en ligne	Vous pouvez ajouter autant de politiques en ligne que désiré à un utilisateur, un rôle ou un groupe IAM. Toutefois, la taille totale de la politique agrégée (la somme de toutes les politiques en ligne) par entité ne peut pas dépasser les limites suivantes : • La taille de la politique d'utilisateur ne peut pas dépasser 2 048 caractères. • La taille de la politique de rôle ne peut pas dépasser 10 240 caractères. • La taille de la politique de groupe ne peut pas dépasser 5 120 caractères. **Note** IAM ne compte pas les espaces lors du calcul de la taille d'une politique au regard de ces limites.
Pour les politiques gérées	• La taille de chaque politique gérée ne peut pas dépasser 6 144 caractères. **Note** IAM ne compte pas les espaces lors du calcul de la taille d'une politique au regard de cette limite.
Nom du groupe	128 caractères
Nom du profil d'instance	128 caractères

Description	Limite
Mot de passe pour un profil de connexion	1–128 caractères
Chemin	512 caractères
Nom de la politique	128 caractères
Nom de rôle	64 caractères **Important** Si vous avez l'intention d'utiliser un rôle avec la fonctionnalité Switch Role dans le AWS Management Console, alors la combinaison `Path` `RoleName` ne doit pas dépasser 64 caractères.

Description	Limite
Durée de la session de rôle	12 heures Lorsque vous assumez un rôle depuis l'API AWS CLI or, vous pouvez utiliser le paramètre `duration-seconds` CLI ou le paramètre `DurationSeconds` API pour demander une session de rôle plus longue. Vous pouvez spécifier une valeur comprise entre 900 secondes (15 minutes) et la durée de session maximale pour le rôle, qui peut varier de 1 à 12 heures. Si vous ne spécifiez pas de valeur pour le paramètre `DurationSeconds` , vos informations d'identification de sécurité sont valides pendant une heure. Les utilisateurs IAM qui changent de rôle dans la console se voient accorder la durée de session maximale ou le temps restant dans la session de l'utilisa teur IAM, selon la durée la plus courte. Le paramètre de durée maximale de session ne limite pas les sessions endossées par les services AWS . Pour savoir comment afficher la valeur maximale pour votre rôle, veuillez consulter Affichage du paramètre de durée de session maximale pour un rôle.
Nom de la session de rôle	64 caractères

Description	Limite
Politiques de session du rôle	• La taille du document de politique JSON transmis et tous les caractères ARN de politique gérée transmis combinés ne peut pas dépasser 2 048 caractères. • Vous pouvez transmettre un maximum de 10 ARN de politique gérée lorsque vous créez une session. • Vous pouvez transmettre un seul document de politique JSON lorsque vous créez par programmation une session temporaire pour un rôle ou un utilisateur fédéré. • En outre, une AWS conversion compresse les politiques de session et les balises de session adoptées dans un format binaire compressé doté d'une limite distincte. L'élément de réponse `PackedPolicySize` indique en pourcentage la proximité des stratégies et des balises de votre requête par rapport à la limite de taille supérieure. • Nous vous recommandons de transmettre les politiques de session à l'aide de l' AWS API AWS CLI or. Ils AWS Management Console peuvent ajouter des informations de session de console supplémentaires à la politique compressée.

Description	Limite
Balises de session du rôle	• Les balises de session doivent respecter la limite de la clé de balise de 128 caractère s et la limite de la valeur de balise de 256 caractères. • Vous pouvez transmettre jusqu'à 50 balises de session. • Une AWS conversion compresse les politiques de session et les balises de session adoptées dans un format binaire compressé doté d'une limite distincte. Vous pouvez transmettre des balises de session à l'aide de l' AWS API AWS CLI or. L'élément de réponse `PackedPolicySize` indique en pourcentage la proximité des stratégies et des balises de votre requête par rapport à la limite de taille supérieure.
Réponse d'authentification SAML codée en base64	100 000 caractères Cette limite de caractères s'applique à l'opérati on CLI `assume-role-with-saml` ou d'API `AssumeRoleWithSAML`.
Clé de balise	128 caractères Cette limite de caractères s'applique aux balises des ressources IAM et des balises de session.

Description	Limite
Valeur de balise	256 caractères Cette limite de caractères s'applique aux balises des ressources IAM et des balises de session. Les valeurs de balise peuvent être vides, ce qui signifie que les valeurs de balise peuvent ne comporter aucun caractère.
ID uniques créés par IAM	128 caractères Par exemple : • ID d'utilisateur commençant par `AIDA` • ID de groupe commençant par `AGPA` • ID de rôle commençant par `AROA` • ID de politique gérée commençant par `ANPA` • ID de stratégie gérée commençant par `ASCA` **Note** Cette liste n'est pas exhaustive. Elle ne garantit pas non plus que les ID d'un certain type commencent uniquement par la combinaison de lettres indiquée.
Nom utilisateur	64 caractères

Points de terminaison de VPC d’Interface

Si vous utilisez Amazon Virtual Private Cloud (Amazon VPC) pour héberger vos AWS ressources, vous pouvez établir une connexion privée entre votre VPC et (). AWS Security Token Service AWS STS Vous pouvez utiliser cette connexion pour AWS STS communiquer avec vos ressources dans votre VPC sans passer par l'Internet public.

Amazon VPC est un AWS service que vous pouvez utiliser pour lancer AWS des ressources dans un réseau virtuel que vous définissez. Avec un VPC, vous contrôlez des paramètres réseau, tels que la plage d'adresses IP, les sous-réseaux, les tables de routage et les passerelles réseau. Pour connecter votre VPC à AWS STS, vous définissez un point de terminaison VPC d'interface pour. AWS STS Le point de terminaison fournit une connectivité fiable et évolutive AWS STS sans nécessiter de passerelle Internet, d'instance de traduction d'adresses réseau (NAT) ou de connexion VPN. Pour de plus amples informations, veuillez consulter Qu'est-ce qu'Amazon VPC ? dans le Guide de l'utilisateur Amazon VPC.

Les points de terminaison VPC d'interface sont alimentés par AWS PrivateLink une AWS technologie qui permet une communication privée entre les AWS services à l'aide d'une interface Elastic Network avec des adresses IP privées. Pour plus d'informations, reportez-vous AWS PrivateLink à la section AWS Services.

Les informations suivantes sont destinées aux utilisateurs d'Amazon VPC. Pour de plus amples informations, veuillez consulter Mise en route avec Amazon VPC dans le Guide de l'utilisateur Amazon VPC.

Disponibilité

AWS STS prend actuellement en charge les points de terminaison VPC dans les régions suivantes :

- USA Est (Virginie du Nord)
- USA Est (Ohio)
- USA Ouest (Californie du Nord)
- USA Ouest (Oregon)
- Afrique (Le Cap)
- Asie-Pacifique (Hong Kong)
- Asie-Pacifique (Hyderabad)
- Asie-Pacifique (Jakarta)
- Asie-Pacifique (Melbourne)
- Asie-Pacifique (Mumbai)
- Asie-Pacifique (Osaka)
- Asia Pacific (Seoul)
- Asie-Pacifique (Singapour)

- Asie-Pacifique (Sydney)
- Asie-Pacifique (Tokyo)
- Canada (Centre)
- Canada Ouest (Calgary)
- Chine (Beijing)
- China (Ningxia)
- Europe (Francfort)
- Europe (Irlande)
- Europe (Londres)
- Europe (Milan)
- Europe (Paris)
- Europe (Espagne)
- Europe (Stockholm)
- Europe (Zurich)
- Israël (Tel Aviv)
- Moyen-Orient (Bahreïn)
- Moyen-Orient (EAU)
- Amérique du Sud (São Paulo)
- AWS GovCloud (USA Est)
- AWS GovCloud (US-Ouest)

Créez un point de terminaison VPC pour AWS STS

Pour commencer à utiliser AWS STS avec votre VPC, créez un point de terminaison VPC d'interface pour. AWS STS Pour plus d'informations, consultez la section Accès à un AWS service à l'aide d'un point de terminaison VPC d'interface dans le guide de l'utilisateur Amazon VPC.

Après avoir créé le point de terminaison VPC, vous devez utiliser le point de terminaison régional correspondant pour envoyer vos AWS STS demandes. AWS STS vous recommande d'utiliser à la fois les `setEndpoint` méthodes `setRegion` et pour passer des appels à un point de terminaison régional. Vous pouvez utiliser la méthode `setRegion` de manière autonome pour répondre à des

régions activées manuellement, par exemple Asie-Pacifique (Hong Kong). Dans ce cas, les appels sont dirigés vers le point de terminaison régional STS. Pour savoir comment activer manuellement une région, veuillez consulter Gestion des régions AWS dans le Références générales AWS. Si vous utilisez la méthode `setRegion` uniquement pour les régions activées par défaut, les appels sont dirigés vers le point de terminaison global `https://sts.amazonaws.com` .

Lorsque vous utilisez des points de terminaison régionaux, AWS STS appelez d'autres AWS services en utilisant des points de terminaison publics ou des points de terminaison VPC d'interface privée, selon ceux utilisés. Supposons, par exemple, que vous avez créé un point de terminaison VPC d'interface pour les ressources situées dans votre VPC AWS STS et que vous avez déjà demandé des informations d'identification temporaires AWS STS auprès de celles-ci. Dans ce cas, ces informations d'identification commencent à circuler via le point de terminaison VPC d'interface par défaut. Pour plus d'informations sur l'envoi de demandes régionales à AWS STS l'aide deGérer AWS STS dans un Région AWS.

AWS services qui fonctionnent avec IAM

Les AWS services répertoriés ci-dessous sont regroupés par ordre alphabétique et incluent des informations sur les fonctionnalités IAM qu'ils prennent en charge :

- Service : vous pouvez choisir le nom d'un service pour consulter la AWS documentation relative à l'autorisation IAM et à l'accès à ce service.
- Actions : Vous pouvez spécifier des actions individuelles dans une politique. Si le service ne prend pas en charge cette fonction, Toutes les actions est sélectionné dans l'éditeur visuel. Dans un document de politique JSON, vous devez utiliser le caractère générique `*` dans l'élément `Action`. Pour obtenir la liste des actions de chaque service, consultez la section Actions, ressources et clés de condition pour les AWS services.
- Autorisations de niveau ressource : Vous pouvez utiliser des ARN pour spécifier des ressources spécifiques dans la politique. Si le service ne prend pas en charge cette fonction, Toutes les ressources est sélectionné dans l'éditeur visuel de politique. Dans un document de politique JSON, vous devez utiliser le caractère générique `*` dans l'élément `Resource`. Certaines actions, comme les actions `List*`, ne prennent pas en charge la spécification d'un ARN, car elles sont conçues pour renvoyer plusieurs ressources. Si un service prend en charge cette fonctionnalité pour certaines ressources, mais pas pour d'autres, cela est indiqué par Partiel dans le tableau. Consultez la documentation de ce service pour plus d'informations.
- Politiques basées sur les ressources : Vous pouvez attacher des politiques basées sur les ressources à une ressource du service. Les politiques basées sur les ressources incluent un

élément `Principal` pour spécifier les identités IAM qui peuvent accéder à cette ressource. Pour plus d'informations, consultez Politiques basées sur l'identité et Politiques basées sur une ressource.

- ABAC (Autorisation basée sur les balises) : pour contrôler l'accès basé sur des balises, vous devez fournir les informations de balises dans l'élément de condition d'une politique utilisant les clés de condition `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` ou `aws:TagKeys`. Si un service prend en charge les trois clés de condition pour tous les types de ressources, alors la valeur pour ce service est Oui. Si un service prend en charge les trois clés de condition pour certains types de ressources uniquement, la valeur est Partielle. Pour de plus amples informations sur la définition d'autorisations basées sur des attributs tels que des balises, veuillez consulter À quoi sert ABAC ? AWS. Pour accéder à un didacticiel décrivant les étapes de configuration de l'ABAC, consultez Use attribute-based access control (ABAC). (Utilisation du contrôle d'accès basé sur les attributs).
- Informations d'identification temporaires : vous pouvez utiliser les informations d'identification à court terme que vous obtenez lorsque vous vous connectez à l'aide d'IAM Identity Center, que vous changez de rôle dans la console ou que vous générez AWS STS à l'aide de l' AWS API AWS CLI or. Vous pouvez accéder aux services avec une valeur No uniquement en utilisant vos informations d'identification utilisateur IAM à long terme. Cela inclut un nom d'utilisateur et un mot de passe ou vos clés d'accès utilisateur. Pour plus d'informations, veuillez consulter Informations d'identification de sécurité temporaires dans IAM.
- Rôles liés à un service : Un rôle lié à un service est un type particulier de rôle qui donne à un service l'autorisation d'accéder aux ressources d'autres services en votre nom. Cliquez sur le lien Oui ou Partiel pour afficher la documentation pour les services qui prennent en charge ces rôles. Cette colonne n'indique pas si le service utilise des rôles de service standard. Pour plus d'informations, veuillez consulter Utilisation des rôles liés à un service.
- Autres informations : Si un service ne prend pas entièrement en charge une fonction, vous pouvez consulter les notes de bas de page d'une entrée pour afficher les limitations et les liens vers les informations connexes.

Services qui fonctionnent avec IAM

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Account Management	Oui	Oui	Non	Non	Oui	Non
AWS Activate Console	Oui	Non	Non	Non	Oui	Non
AWS Amplify Administrateur	Oui	Oui	Non	Non	Oui	Non
AWS Amplify	Oui	Oui	Non	Partielle	Oui	Non
AWS Amplify Générateur d'interface utilisateur	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
API Apache Kafka pour les clusters Amazon MSK	 Oui	 Oui	 Non	 Non	 Oui	 Non
Amazon API Gateway	 Oui	 Oui	 Oui	 Non	 Oui	 Oui
Gestion Amazon API Gateway	 Oui	 Oui	 Non	 Oui	 Oui	 Non
Gestion Amazon API Gateway V2	 Oui	 Oui	 Non	 Oui	 Oui	 Non
AWS App2Container	 Oui	 Non	 Non	 Non	 Oui	 Non
AWS AppConfig	 Oui	 Oui	 Non	 Oui	 Oui	 Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS AppFabric	Oui	Oui	Non	Oui	Oui	Non
Amazon AppFlow	Oui	Oui	Non	Oui	Oui	Non
Amazon AppIntegrations	Oui	Oui	Non	Oui	Oui	Oui
Application Auto Scaling	Oui	Oui	Non	Oui	Oui	Oui
AWS Profileur des coûts d'application	Oui	Non	Non	Non	Oui	Non
AWS Arsenal de découverte d'applications	Oui	Non	Non	Non	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Application Discovery Service	Oui	Non	Non	Non	Oui	Oui
AWS Application Migration Service	Oui	Oui	Non	Oui	Oui	Oui
AWS Service de transformation des applications	Oui	Non	Non	Non	Oui	Non
AWS App Mesh	Oui	Oui	Non	Oui	Oui	Oui
AWS App Mesh Version préliminaire	Oui	Oui	Non	Non	Oui	Oui
AWS App Runner	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon AppStream 2.0	Oui	Oui	Non	Oui	Oui	Non
AWS AppSync	Oui	Oui	Non	Oui	Oui	Non
AWS Artifact	Oui	Oui	Non	Non	Oui	Non
Amazon Athena	Oui	Oui	Non	Oui	Oui	Non
AWS Audit Manager	Oui	Oui	Non	Oui	Oui	Oui
AWS Auto Scaling	Oui	Non	Non	Non	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Échange de données B2B	Oui	Oui	Non	Oui	Oui	Non
AWS Backup	Oui	Oui	Oui	Oui	Oui	Oui
AWS Backup Passerelle	Oui	Oui	Non	Oui	Oui	Non
AWS Backup rangement	Oui	Non	Non	Non	Oui	Non
AWS Batch	Oui	Partielle	Non	Oui	Oui	Oui
Amazon Bedrock	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Billing and Cost Management	Oui	Non	Non	Non	Oui	Oui
AWS Billing and Cost Management Exportations de données	Oui	Oui	Non	Oui	Oui	Non
AWS Billing Conductor	Oui	Oui	Non	Oui	Oui	Non
Amazon Braket	Oui	Oui	Non	Oui	Oui	Oui
AWS Service du budget	Oui	Oui	Non	Non	Non	Non
AWS BugBust	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Certificate Manager (ACM)	Oui	Oui	Non	Oui	Oui	Oui
AWS Chatbot	Oui	Oui	Non	Non	Oui	Oui
Amazon Chime	Oui	Oui	Non	Oui	Oui	Oui
AWS Clean Rooms	Oui	Oui	Non	Oui	Oui	Non
AWS Clean Rooms ML	Oui	Oui	Non	Oui	Oui	Non
AWS Client VPN	Oui	Oui	Non	Non	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Cloud9	 Oui	 Oui	 Oui	 Oui	 Oui	 Oui
AWS Cloud API de contrôle	 Oui	 Non	 Non	 Non	 Oui	 Non
Amazon Cloud Directory	 Oui	 Oui	 Non	 Non	 Oui	 Non
AWS CloudFormation	 Oui	 Oui	 Non	 Oui	 Oui	 Non
Amazon CloudFront	 Oui	 Oui	 Non	 Partielle	 Oui	 Partiel (Informations)

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon CloudFront KeyValueStore	 Oui	 Oui	 Non	 Non	 Oui	 Non
AWS CloudHSM	 Oui	 Oui	 Non	 Oui	 Oui	 Oui
AWS Cloud Map	 Oui	 Oui	 Non	 Oui	 Oui	 Non
Amazon CloudSearch	 Oui	 Oui	 Non	 Non	 Oui	 Non
AWS CloudShell	 Oui	 Oui	 Non	 Non	 Oui	 Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS CloudTrail	Oui	Oui	Partiel (Informations)	Partiel (Informations)	Oui	Oui
AWS CloudTrail Données	Oui	Oui	Non	Oui	Oui	Non
Amazon CloudWatch	Oui	Oui	Non	Oui	Oui	Partiel (Informations)
Informations sur les CloudWatch applications Amazon	Oui	Non	Non	Non	Oui	Non
Signaux CloudWatch d'application Amazon	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon, CloudWatch évidemment	Oui	Oui	Non	Oui	Oui	Non
Amazon CloudWatch Internet Monitor	Oui	Oui	Non	Oui	Oui	Non
Amazon CloudWatch Logs	Oui	Oui	Oui	Partielle	Oui	Oui
Amazon CloudWatch Network Monitor	Oui	Oui	Non	Oui	Oui	Non
Gestionnaire d'accès Amazon CloudWatch Observability	Oui	Oui	Non	Oui	Oui	Non
Amazon CloudWatch RUM	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon CloudWatch Synthetics	Oui	Oui	Non	Oui	Oui	Non
AWS CodeArtifact	Oui	Oui	Oui	Oui	Oui	Non
AWS CodeBuild	Oui	Oui	Oui (Informations)	Partiel (Informations)	Oui	Non
Amazon CodeCatalyst	Oui	Oui	Non	Oui	Oui	Oui
AWS CodeCommit	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS CodeConnections	Oui	Oui	Non	Oui	Oui	Non
AWS CodeDeploy	Oui	Oui	Non	Oui	Oui	Non
AWS CodeDeploy service de commandes hôte sécurisé	Oui	Non	Non	Non	Oui	Non
Amazon CodeGuru Profiler	Oui	Oui	Non	Oui	Oui	Oui
CodeGuru Réviseur Amazon	Oui	Oui	Non	Oui	Oui	Oui
Amazon CodeGuru Security	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS CodePipeline	Oui	Partielle	Non	Oui	Oui	Non
AWS CodeStar	Oui	Partielle	Non	Oui	Oui	Non
AWS CodeStar Connexions	Oui	Oui	Non	Oui	Oui	Oui
AWS CodeStar Notifications	Oui	Oui	Non	Oui	Oui	Oui
Amazon CodeWhisperer	Oui	Oui	Non	Oui	Oui	Oui
Amazon Cognito	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Cognito Sync	Oui	Oui	Non	Non	Oui	Oui
Groupes d'utilisateurs Amazon Cognito	Oui	Oui	Non	Oui	Oui	Oui
Amazon Comprehend	Oui	Oui	Non	Oui	Oui	Non
Amazon Comprehend Medical	Oui	Non	Non	Non	Oui	Non
AWS Compute Optimizer	Oui	Non	Non	Non	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Config	Oui	Partiel (Informations)	Non	Oui	Oui	Oui
Amazon Connect	Oui	Oui	Non	Oui	Oui	Oui
Cas d'Amazon Connect	Oui	Oui	Non	Oui	Oui	Non
Amazon Connect Customer Profiles	Oui	Oui	Non	Oui	Oui	Oui
Communications sortantes à volume important Amazon Connect	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Connect Voice ID	 Oui	 Oui	 Non	 Oui	 Oui	 Non
AWS Console Mobile Application	 Oui	 Oui	 Non	 Non	 Oui	 Non
AWS Facturation consolidée	 Oui	 Non	 Non	 Non	 Oui	 Non
AWS Catalogue de contrôle	 Oui	 Oui	 Non	 Non	 Oui	 Non
AWS Control Tower	 Oui	 Oui	 Non	 Non	 Oui	 Non
AWS Cost and Usage Report	 Oui	 Oui	 Non	 Non	 Oui	 Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Cost Explorer	Oui	Oui	Non	Oui	Oui	Non
Hub d'optimisation des coûts	Oui	Non	Non	Non	Oui	Non
Service de vérification des clients AWS	Oui	Non	Non	Non	Oui	Non
AWS Database Migration Service	Oui	Oui	Non (Informations)	Oui	Oui	Oui
Database Query Metadata Service	Oui	Non	Non	Non	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Data Exchange	Oui	Oui	Non	Oui	Oui	Non
Amazon Data Lifecycle Manager	Oui	Oui	Non	Oui	Oui	Non
AWS Data Pipeline	Oui	Oui	Non	Partielle	Oui	Non
AWS DataSync	Oui	Oui	Non	Oui	Oui	Oui
Amazon DataZone	Oui	Non	Non	Non	Oui	Non
AWS Deadline Cloud	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS DeepComposer	Oui	Oui	Non	Oui	Oui	Non
AWS DeepRacer	Oui	Oui	Non	Oui	Oui	Oui
Amazon Detective	Oui	Oui	Non	Oui	Oui	Non
AWS Device Farm	Oui	Oui	Non	Oui	Oui	Oui
Amazon DevOps Guru	Oui	Oui	Non	Non	Oui	Oui
Outils de diagnostic AWS	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Direct Connect	 Oui	 Oui	 Non	 Oui	 Oui	 Oui
AWS Directory Service	 Oui	 Oui	 Non	 Oui	 Oui	 Non
Clusters élastiques Amazon DocumentDB	 Oui	 Oui	 Non	 Oui	 Oui	 Oui
Amazon DynamoDB Accelerator (DAX)	 Oui	 Oui	 Non	 Non	 Oui	 Oui
Amazon DynamoDB	 Oui	 Oui	 Oui	 Non	 Oui	 Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Elastic Compute Cloud (Amazon EC2)	Oui	Partielle	Non	Oui	Oui	Partiel (Informations)
Amazon EC2 Auto Scaling	Oui	Oui	Non	Oui	Oui	Oui
EC2 Image Builder	Oui	Oui	Non	Oui	Oui	Oui
Amazon EC2 Instance Connect	Oui	Oui	Non	Non	Oui	Oui
Amazon ElastiCache	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Elastic Beanstalk	Oui	Partielle	Non	Oui	Oui	Oui
Amazon Elastic Block Store (Amazon EBS)	Oui	Partielle	Non	Oui	Oui	Non
Amazon Elastic Container Registry (Amazon ECR)	Oui	Oui	Oui	Oui	Oui	Oui
Amazon Elastic Container Registry Public (Amazon ECR Public)	Oui	Oui	Non	Oui	Oui	Non
Amazon Elastic Container Service (Amazon ECS)	Oui	Partiel (Informations)	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Elastic Disaster Recovery	Oui	Oui	Non	Oui	Oui	Oui
Amazon Elastic File System (Amazon EFS)	Oui	Oui	Oui	Partielle	Oui	Oui
Amazon Elastic Inference	Oui	Oui	Non	Non	Oui	Non
Amazon Elastic Kubernetes Service (Amazon EKS)	Oui	Oui	Non	Oui	Oui	Oui
Authentification Amazon Elastic Kubernetes Service (Amazon EKS)	Oui	Oui	Non	Non	Oui	Non
AWS Elastic Load Balancing	Oui	Partielle	Non	Partielle	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Elastic Transcoder	Oui	Oui	Non	Non	Oui	Non
AWS Service d'activation des appliances élémentaires et des logiciels	Oui	Oui	Non	Oui	Oui	Non
AWS Appliances et logiciels Elemental	Oui	Oui	Non	Oui	Oui	Non
AWS Elemental MediaConnect	Oui	Oui	Non	Non	Oui	Oui
AWS Elemental MediaConvert	Oui	Oui	Non	Oui	Oui	Non
AWS Elemental MediaLive	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Elemental MediaPackage	Oui	Oui	Non	Oui	Oui	Partiel (Informations)
AWS Elemental MediaPackage V2	Oui	Oui	Non	Oui	Oui	Non
AWS Elemental MediaPackage VOD	Oui	Oui	Non	Oui	Oui	Partiel (Informations)
AWS Elemental MediaStore	Oui	Oui	Oui	Oui	Oui	Non
AWS Elemental MediaTailor	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Dossiers de support élémentaires	Oui	Non	Non	Non	Oui	Non
AWS Contenu de support élémentaire	Oui	Non	Non	Non	Oui	Non
Amazon EMR	Oui	Oui	Non	Oui	Oui	Oui
Amazon EMR on EKS	Oui	Oui	Non	Oui	Oui	Oui
Amazon EMR sans serveur	Oui	Oui	Non	Oui	Oui	Oui
AWS Résolution de l'entité	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon EventBridge	Oui	Oui	Oui	Oui	Oui	Non
Amazon EventBridge Pipes	Oui	Oui	Non	Oui	Oui	Non
Amazon EventBridge Scheduler	Oui	Oui	Non	Oui	Oui	Non
EventBridge Schémas Amazon	Oui	Oui	Oui	Oui	Oui	Non
AWS Service d'injection de défauts	Oui	Oui	Non	Oui	Oui	Oui
Amazon FinSpace	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon FinSpace API	Oui	Oui	Non	Non	Oui	Non
AWS Firewall Manager	Oui	Oui	Non	Oui	Oui	Partielle
Fleet Hub for AWS IoT Device Management	Oui	Oui	Non	Oui	Oui	Non
Amazon Forecast	Oui	Oui	Non	Oui	Oui	Non
Amazon Fraud Detector	Oui	Oui	Non	Oui	Oui	Non
FreeRTOS	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Niveau gratuit	Oui	Non	Non	Non	Oui	Non
Amazon FSx	Oui	Oui	Non	Oui	Oui	Oui
Amazon GameLift	Oui	Oui	Non	Oui	Oui	Non
AWS Global Accelerator	Oui	Oui	Non	Oui	Oui	Oui
AWS Glue	Oui	Oui	Oui	Partielle	Oui	Non
AWS Glue DataBrew	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Ground Station	Oui	Oui	Non	Oui	Oui	Oui
Étiquetage Amazon Ground Truth	Oui	Non	Non	Non	Oui	Non
Amazon GuardDuty	Oui	Oui	Non	Oui	Oui	Oui
AWS Health API et notifications	Oui	Oui	Non	Non	Oui	Non
AWS HealthImaging	Oui	Oui	Non	Oui	Oui	Non
AWS HealthLake	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS HealthOmics	Oui	Oui	Non	Oui	Oui	Non
Amazon Honeycode	Oui	Oui	Non	Non	Oui	Non
AWS IAM Identity Center	Oui	Oui	Non	Partielle	Oui	Oui
Répertoire d'IAM Identity Center	Oui	Non	Non	Non	Oui	Non
Magasin d'identités d'IAM Identity Center	Oui	Oui	Non	Non	Oui	Non
Service OIDC d'IAM Identity Center	Oui	Oui	Non	Non	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Identity and Access Management (JE SUIS)	Oui	Oui	Partiel (Informations)	Partiel (Informations)	Partiel (Informations)	Non
AWS Identity and Access Management Analyseur d'accès	Oui	Oui	Non	Oui	Oui	Partielle
AWS Identity and Access Management Des rôles n'importe où	Oui	Oui	Non	Oui	Oui	Oui
AWS Authentification du magasin d'identités	Oui	Non	Non	Non	Oui	Non
AWS Synchronisation des identités	Oui	Oui	Non	Non	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Import/Export	Oui	Non	Non	Non	Oui	Non
Amazon Inspector	Oui	Oui	Non	Oui	Oui	Oui
Amazon Inspector Classic	Oui	Non	Non	Non	Oui	Oui
Amazon InspectorScan	Oui	Non	Non	Non	Oui	Non
Amazon Interactive Video Service	Oui	Oui	Non	Oui	Oui	Oui
Service de vidéo interactive Amazon Chat	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Facturation AWS	 Oui	 Non	 Non	 Non	 Oui	 Non
AWS IoT 1-Click	 Oui	 Oui	 Non	 Oui	 Oui	 Non
AWS IoT Analytics	 Oui	 Oui	 Non	 Oui	 Oui	 Non
AWS IoT	 Oui	 Oui	 Partiel (Informations)	 Oui	 Oui	 Non
AWS IoT Core Conseiller en matière d'appareils	 Oui	 Oui	 Non	 Oui	 Oui	 Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS IoT Testeur d'appareils	Oui	Non	Non	Non	Oui	Non
AWS IoT Events	Oui	Oui	Non	Oui	Oui	Non
AWS IoT FleetWise	Oui	Oui	Non	Oui	Oui	Non
AWS IoT Greengrass	Oui	Oui	Non	Oui	Oui	Non
AWS IoT Greengrass V2	Oui	Oui	Non	Partielle	Oui	Non
AWS IoT Emplois DataPlane	Oui	Oui	Non	Non	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS IoT RoboRunner	Oui	Oui	Non	Non	Oui	Non
AWS IoT SiteWise	Oui	Oui	Non	Oui	Oui	Oui
AWS IoT TwinMaker	Oui	Oui	Non	Oui	Oui	Oui
AWS IoT Wireless	Oui	Oui	Non	Oui	Oui	Non
AWS IQ	Oui	Oui	Non	Non	Oui	Oui
AWS Autorisations IQ	Oui	Oui	Non	Non	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Kendra	Oui	Oui	Non	Oui	Oui	Non
Amazon Kendra Intelligent Ranking (Classement intelligent Amazon Kendra)	Oui	Oui	Non	Oui	Oui	Non
AWS Key Management Service (AWS KMS)	Oui	Oui	Oui	Oui	Oui	Oui
Amazon Keyspaces (pour Apache Cassandra)	Oui	Oui	Non	Oui	Oui	Oui
Service géré Amazon pour Apache Flink	Oui	Oui	Non	Oui	Oui	Non
Service géré Amazon pour Apache Flink V2	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Data Firehose	Oui	Oui	Non	Oui	Oui	Non
Amazon Kinesis Data Streams	Oui	Oui	Oui	Non	Oui	Non
Amazon Kinesis Video Streams	Oui	Oui	Non	Oui	Oui	Non
AWS Lake Formation	Oui	Non	Non	Non	Oui	Oui
AWS Lambda	Oui	Oui	Oui	Partiel (Informations)	Oui	Partiel (Informations)

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Launch Wizard	Oui	Non	Non	Non	Oui	Non
Amazon Lex	Oui	Oui	Non	Oui	Oui	Oui
Amazon Lex V2	Oui	Oui	Oui	Oui	Oui	Oui
AWS License Manager	Oui	Oui	Non	Oui	Oui	Oui
AWS License Manager Gestionnaire des abonnements Linux	Oui	Non	Non	Non	Oui	Non
AWS License Manager Abonnements d'utilisateurs	Oui	Non	Non	Non	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Lightsail	Oui	Partiel (Informations)	Non	Partiel (Informations)	Oui	Oui
Amazon Location Service	Oui	Oui	Non	Oui	Oui	Non
Amazon Lookout for Equipment	Oui	Oui	Non	Oui	Oui	Non
Amazon Lookout for Metrics	Oui	Oui	Non	Oui	Oui	Non
Amazon Lookout for Vision	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Machine Learning	Oui	Oui	Non	Non	Oui	Non
Amazon Macie	Oui	Oui	Non	Oui	Oui	Oui
AWS Mainframe Modernization	Oui	Oui	Non	Oui	Oui	Oui
AWS Mainframe Modernization Tests d'applications	Oui	Oui	Non	Oui	Oui	Non
Amazon Managed Blockchain	Oui	Oui	Non	Oui	Oui	Non
Amazon Managed Blockchain Query	Oui	Non	Non	Non	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Managed Grafana	Oui	Oui	Non	Oui	Oui	Oui
Amazon Managed Service for Prometheus	Oui	Oui	Non	Oui	Oui	Non
Amazon Managed Streaming for Apache Kafka (MSK)	Oui	Oui	Partiel (Informations)	Oui	Oui	Oui
Amazon Managed Streaming for Kafka Connect	Oui	Oui	Non	Non	Oui	Oui
Amazon Managed Workflows for Apache Airflow	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Marketplace	Oui	Non	Non	Non	Oui	Oui
AWS Marketplace Catalogue	Oui	Oui	Non	Oui	Oui	Non
AWS Marketplace Commerce Analytics	Oui	Non	Non	Non	Non	Non
Service de déploiement AWS Marketplace	Oui	Oui	Non	Oui	Oui	Non
AWS Marketplace Découverte	Oui	Non	Non	Non	Oui	Non
AWS Marketplace Entitlement Service	Oui	Non	Non	Non	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Marketplace Service de création d'images	Oui	Non	Non	Non	Oui	Non
Portail de gestion AWS Marketplace	Oui	Non	Non	Non	Oui	Non
AWS Marketplace Metering Service	Oui	Non	Non	Non	Oui	Non
AWS Marketplace Marketplace privée	Oui	Non	Non	Non	Oui	Non
AWS Marketplace Intégration des systèmes d'approvisionnement	Oui	Non	Non	Non	Oui	Non
AWS Marketplace Rapports sur les vendeurs	Oui	Oui	Non	Non	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Marketplace Informations sur les fournisseurs	Oui	Oui	Non	Oui	Oui	Non
Amazon Mechanical Turk	Oui	Non	Non	Non	Oui	Non
Amazon MediaImport	Oui	Non	Non	Non	Non	Non
Amazon MemoryDB for Redis	Oui	Oui	Non	Oui	Oui	Oui
Amazon Message Delivery Service	Oui	Non	Non	Non	Oui	Non
Service Amazon Message Gateway	Oui	Non	Non	Non	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Microservice Extractor for .NET	Oui	Non	Non	Non	Oui	Non
AWS Crédits du programme Migration Acceleration	Oui	Oui	Non	Non	Oui	Non
AWS Migration Hub	Oui	Oui	Non	Non	Oui	Oui
Orchestrateur AWS Migration Hub	Oui	Oui	Non	Oui	Oui	Oui
AWS Migration Hub Refactor Spaces	Oui	Oui	Oui	Oui	Oui	Oui
Recommandations de stratégie AWS Migration Hub	Oui	Non	Non	Non	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Monitron	Oui	Oui	Non	Oui	Oui	Oui
Amazon MQ	Oui	Oui	Non	Oui	Oui	Oui
Amazon Neptune	Oui	Oui	Non	Non	Oui	Oui
Amazon Neptune Analytics	Oui	Oui	Non	Oui	Oui	Non
AWS Network Firewall	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Network Manager	Oui	Oui	Non	Oui	Oui	Oui (Informations)
AWS Network Manager Discuter	Oui	Non	Non	Non	Oui	Non
Amazon Nimble Studio	Oui	Oui	Non	Oui	Oui	Non
Amazon One Enterprise	Oui	Oui	Non	Oui	Oui	Non
OpenSearchIngestion d'Amazon	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon OpenSearch sans serveur	Oui	Oui	Non	Oui	Oui	Oui
Amazon OpenSearch Service	Oui	Oui	Oui	Oui	Oui	Oui
AWS OpsWorks	Oui	Oui	Non	Non	Oui	Non
AWS OpsWorks Gestion de configuration	Oui	Oui	Non	Non	Oui	Non
AWS Organizations	Oui	Oui	Non	Oui	Non	Oui
AWS Outposts	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Panorama	 Oui	 Oui	 Non	 Oui	 Oui	 Oui
AWS Partner Gestion centralisée des comptes	 Oui	 Non	 Non	 Non	 Oui	 Non
AWS Payment Cryptography	 Oui	 Oui	 Non	 Oui	 Oui	 Non
AWS Paiements	 Oui	 Non	 Non	 Non	 Oui	 Non
AWS Performance Insights	 Oui	 Oui	 Non	 Non	 Oui	 Non
Amazon Personalize	 Oui	 Oui	 Non	 Non	 Oui	 Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Pinpoint	Oui	Oui	Non	Oui	Oui	Non
Service de messagerie Amazon Pinpoint	Oui	Oui	Non	Oui	Oui	Non
Service de messages SMS et vocaux Amazon Pinpoint	Oui	Non	Non	Non	Oui	Non
Service de messages SMS et vocaux Amazon Pinpoint V2	Oui	Oui	Non	Oui	Oui	Non
Amazon Polly	Oui	Oui	Non	Non	Oui	Non
AWS Price List	Oui	Non	Non	Non	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS 5G privée	Oui	Oui	Non	Oui	Oui	Non
AWS Private CA Connector for Active Directory (français non garanti)	Oui	Oui	Non	Oui	Oui	Non
AWS Private CA Connecteur pour SCEP	Oui	Oui	Non	Oui	Oui	Non
AWS Private Certificate Authority (AWS Private CA)	Oui	Oui	Oui	Oui	Oui	Non
AWS Proton	Oui	Oui	Non	Oui	Oui	Oui
AWS Console de bons de commande	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Q Business	Oui	Oui	Non	Oui	Oui	Oui
Applications Amazon Q Business Q	Oui	Oui	Non	Non	Oui	Non
Développeur Amazon Q	Oui	Non	Non	Non	Oui	Oui
Amazon Q in Connect	Oui	Oui	Non	Oui	Oui	Non
Amazon Quantum Ledger Database (Amazon QLDB)	Oui	Oui	Non	Oui	Oui	Non
Amazon QuickSight	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon RDS Data API	Oui	Oui	Non	Non	Oui	Non
Authentification IAM Amazon RDS	Oui	Oui	Non	Non	Oui	Non
AWS Corbeille	Oui	Oui	Non	Oui	Oui	Non
Amazon Redshift	Oui	Oui	Non	Oui	Oui	Oui
API de données Amazon Redshift	Oui	Oui	Non	Non	Oui	Non
Amazon Redshift sans serveur	Oui	Oui	Oui	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Rekognition	Oui	Oui	Partiel (Informations)	Oui	Oui	Non
Amazon Relational Database Service (Amazon RDS) (Informations)	Oui	Oui	Non	Oui	Oui	Oui
AWS re:Post Privé	Oui	Oui	Non	Oui	Oui	Oui
AWS Resilience Hub	Oui	Oui	Non	Oui	Oui	Non
AWS Resource Access Manager (AWS RAM)	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Explorateur de ressources AWS	Oui	Oui	Non	Oui	Oui	Oui
AWS Resource Groups	Oui	Oui	Non	Oui	Partiel (Informations)	Non
AWS Resource Groups Tagging API	Oui	Non	Non	Non	Oui	Non
Portail Amazon RHEL Knowledgebase	Oui	Non	Non	Non	Oui	Non
AWS RoboMaker	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Route 53	Oui	Oui	Non	Non	Oui	Non
Amazon Route 53 Application Recovery Controller – Changement de zone	Oui	Oui	Non	Non	Oui	Non
Amazon Route 53 Domaines	Oui	Non	Non	Non	Non	Non
Profils Amazon Route 53	Oui	Oui	Non	Oui	Oui	Non
Amazon Route 53 Recovery Cluster	Oui	Oui	Non	Non	Oui	Non
Configuration d'Amazon Route 53 Recovery Controls	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Route 53 Recovery Readiness	Oui	Oui	Non	Oui	Oui	Oui
Amazon Route 53 Resolver	Oui	Oui	Non	Oui	Oui	Oui
Amazon S3 Express	Oui	Oui	Non	Non	Oui	Non
Amazon S3 Glacier	Oui	Oui	Oui	Oui	Oui	Non
Amazon SageMaker	Oui	Oui	Non	Oui	Oui	Partiel (Informations)

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Fonctionnalités SageMaker géospatiales d'Amazon	Oui	Oui	Non	Oui	Oui	Non
Amazon SageMaker Ground Truth Synthetic	Oui	Non	Non	Non	Oui	Non
Savings Plans AWS	Oui	Oui	Non	Oui	Oui	Non
AWS Secrets Manager	Oui	Oui	Oui	Oui	Oui	Non
AWS Security Hub	Oui	Oui	Non	Oui	Oui	Oui
Amazon Security Lake	Oui	Oui	Non	Non	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Security Token Service (AWS STS)	Oui	Partiel (Informations)	Non	Oui	Partiel (Informations)	Non
AWS Serverless Application Repository	Oui	Oui	Oui	Non	Oui	Non
AWS Service Catalog	Oui	Oui	Non	Oui	Oui	Oui
Service Quotas	Oui	Oui	Non	Oui	Oui	Non
AWS Shield	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Signer	Oui	Oui	Oui	Oui	Oui	Non
AWS Se connecter	Oui	Non	Non	Non	Oui	Non
Amazon SimpleDB	Oui	Oui	Non	Non	Oui	Non
Amazon Simple Email Service - Gestionnaire de messagerie	Oui	Oui	Non	Oui	Oui	Oui
Amazon Simple Email Service (Amazon SES) v2	Oui	Partiel (Informations)	Oui	Oui	Partiel (Informations)	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Simple Notification Service (Amazon SNS)	Oui	Oui	Oui	Oui	Oui	Non
Amazon Simple Queue Service (Amazon SQS)	Oui	Oui	Oui	Partielle	Oui	Non
Amazon Simple Storage Service (Amazon S3)	Oui	Oui	Oui	Partiel (Informations)	Oui	Partiel (Informations)
Amazon Simple Storage Service(Amazon S3) Objet Lambda	Oui	Oui	Non	Non	Oui	Non
Amazon Simple Storage Service (Amazon S3) sur AWS Outposts	Oui	Oui	Oui	Non	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Simple Workflow Service (Amazon SWF)	Oui	Oui	Non	Oui	Oui	Non
AWS SimSpaceTisserand	Oui	Oui	Non	Oui	Oui	Non
AWS Site-to-Site VPN	Oui	Oui	Non	Non	Oui	Oui
AWS Snowball	Oui	Non	Non	Non	Oui	Non
AWS Snowball Edge	Oui	Non	Non	Non	Oui	Non
AWS Snow Device Management	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS SQL Workbench	Oui	Oui	Non	Oui	Oui	Non
AWS Step Functions	Oui	Oui	Non	Oui	Oui	Non
AWS Storage Gateway	Oui	Oui	Non	Oui	Oui	Non
AWS Supply Chain	Oui	Oui	Non	Oui	Oui	Non
AWS Support App in Slack	Oui	Non	Non	Non	Oui	Non
AWS Support	Oui	Non	Non	Non	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Support Plans	Oui	Non	Non	Non	Oui	Non
AWS Support Recommandations	Oui	Non	Non	Non	Oui	Non
AWS Durabilité	Oui	Non	Non	Non	Oui	Non
AWS Systems Manager	Oui	Oui	Non	Oui	Oui	Oui
AWS Systems Manager pour SAP	Oui	Oui	Non	Oui	Oui	Non
AWS Systems Manager GUI Connect	Oui	Non	Non	Non	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Systems Manager Incident Manager	Oui	Oui	Oui	Oui	Oui	Oui
AWS Systems Manager Incident Manager Contacts	Oui	Oui	Oui	Non	Oui	Non
Tag Editor	Oui	Non	Non	Non	Oui	Non
AWS Paramètres fiscaux	Oui	Non	Non	Non	Oui	Non
AWS Générateur de réseaux de télécommunications	Oui	Oui	Non	Oui	Oui	Non
Amazon Textract	Oui	Non	Non	Non	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Timestream	Oui	Oui	Non	Oui	Oui	Non
Amazon Timestream Influxdb	Oui	Oui	Non	Oui	Oui	Oui
AWS API Tiros (pour Reachability Analyzer)	Oui	Non	Non	Non	Non	Non
Amazon Transcribe	Oui	Oui	Non	Oui	Oui	Non
AWS Transfer Family	Oui	Oui	Non	Oui	Oui	Non
Amazon Translate	Oui	Oui	Non	Oui	Oui	Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
AWS Trusted Advisor	 Partiel (Informations)	 Oui	 Non	 Non	 Partielle	 Oui
AWS Notifications aux utilisateurs	 Oui	 Oui	 Non	 Oui	 Oui	 Oui
AWS Contacts pour les notifications des utilisateurs	 Oui	 Oui	 Non	 Oui	 Oui	 Non
AWS Abonnements d'utilisateurs	 Oui	 Non	 Non	 Non	 Oui	 Non
Accès vérifié par AWS	 Oui	 Non	 Non	 Non	 Oui	 Non

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Amazon Verified Permissions	Oui	Oui	Non	Non	Oui	Non
Amazon Virtual Private Cloud (Amazon VPC)	Oui	Partiel (Informations)	Partiel (Informations)	Oui	Oui	Partiel (Informations)
Amazon VPC Lattice	Oui	Oui	Non	Oui	Oui	Non
Amazon VPC Lattice Services	Oui	Oui	Non	Non	Oui	Non
AWS WAF	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Classique AWS WAF	Oui	Oui	Non	Oui	Oui	Oui
AWS WAF Régional	Oui	Oui	Non	Oui	Oui	Oui
AWS Well-Architected Tool	Oui	Oui	Non	Oui	Oui	Non
AWS Wickr	Oui	Oui	Non	Oui	Oui	Non
Amazon WorkDocs	Oui	Non	Non	Non	Oui	Non
Amazon WorkMail	Oui	Oui	Non	Oui	Oui	Oui

Service	Actions	Autorisations de niveau ressource	Politiques basées sur les ressources	ABAC	Informations d'identification temporaires	Rôles liés à un service
Flux de WorkMail messages Amazon	 Oui	 Oui	 Non	 Non	 Oui	 Non
Amazon WorkSpaces	 Oui	 Oui	 Non	 Oui	 Oui	 Non
Navigateur Amazon WorkSpaces Secure	 Oui	 Oui	 Non	 Oui	 Oui	 Oui
Amazon WorkSpaces Thin Client	 Oui	 Oui	 Non	 Oui	 Oui	 Non
AWS X-Ray	 Oui	 Partiel (Informations)	 Non	 Partiel (Informations)	 Oui	 Non

En savoir plus

Amazon CloudFront

CloudFront n'a pas de rôles liés à un service, contrairement à Lambda @Edge. Pour plus d'informations, consultez la section Rôles liés aux services pour Lambda @Edge dans le manuel Amazon Developer Guide. CloudFront

AWS CloudTrail

CloudTrail prend en charge les politiques basées sur les ressources uniquement sur les CloudTrail canaux utilisés pour les intégrations de CloudTrail Lake avec des sources d'événements extérieures à. AWS

CloudTrail prend en charge le contrôle d'accès basé sur des balises pour les magasins et canaux de données d'événements de CloudTrail Lake. CloudTrail ne prend pas en charge les contrôles d'accès basés sur des balises pour les sentiers.

Amazon CloudWatch

CloudWatch les rôles liés à un service ne peuvent pas être créés à l'aide de la fonctionnalité Alarm Actions AWS Management Console, et ne prennent en charge que celle-ci.

AWS CodeBuild

CodeBuild prend en charge le partage de ressources entre comptes en utilisant AWS RAM.

CodeBuild soutient ABAC pour les actions basées sur des projets.

AWS Config

AWS Config prend en charge les autorisations au niveau des ressources pour l'agrégation de données multicomptes et multirégions et les règles. AWS Config Pour obtenir une liste des ressources prises en charge, consultez les sections Regroupement des données de plusieurs comptes et plusieurs régions et Règles AWS Config dans le Guide d'API AWS Config.

AWS Database Migration Service

Vous pouvez créer et modifier des politiques associées aux clés de AWS KMS chiffrement que vous créez pour chiffrer les données migrées vers les points de terminaison cibles pris en charge. Les points de terminaison cibles pris en charge incluent Amazon Redshift et Amazon S3. Pour plus d'informations, consultez les sections Création et utilisation de AWS KMS clés pour chiffrer

les données cibles Amazon Redshift et AWS KMS Création de clés pour chiffrer des objets cibles Amazon S3 AWS Database Migration Service dans le guide de l'utilisateur.

Amazon Elastic Compute Cloud

Les rôles liés au service Amazon EC2 ne peuvent être utilisés que pour les fonctionnalités suivantes : demandes d'instances Spot, demandes de parc d'instances Spot, flotte d'Amazon EC2 et lancement rapide pour les instances Windows.

Amazon Elastic Container Service

Seules quelques actions Amazon ECS prennent en charge les autorisations de niveau ressource.

AWS Elemental MediaPackage

MediaPackage prend en charge les rôles liés aux services pour publier les journaux d'accès des clients CloudWatch , mais pas pour d'autres actions d'API.

AWS Identity and Access Management

IAM prend en charge un seul type de politique basée sur les ressources, nommé politique d'approbation de rôle, qui est attaché à un rôle IAM. Pour plus d’informations, consultez Octroi d'autorisations à un utilisateur pour endosser un rôle.

IAM prend en charge le contrôle d'accès basé sur des balises pour la plupart des ressources IAM. Pour plus d’informations, consultez Balisage des ressources IAM.

Seules quelques actions d'API pour IAM peuvent être appelées avec des informations d'identification temporaires. Pour plus d'informations, consultez Comparaison de vos options d'API.

AWS IoT

Les appareils connectés AWS IoT sont authentifiés à l'aide de certificats X.509 ou à l'aide d'Amazon Cognito Identities. Vous pouvez associer des AWS IoT politiques à un certificat X.509 ou à Amazon Cognito Identity pour contrôler ce que l'appareil est autorisé à faire. Pour plus d'informations, consultez Sécurité et identité pour AWS IoT dans le Manuel du développeur AWS IoT .

AWS Lambda

Lambda prend en charge le contrôle d'accès par attributs (ABAC) pour les actions d'API qui utilisent une fonction Lambda comme ressource requise. Les couches, les mappages de sources d'événements et les ressources de configuration de signature de code ne sont pas pris en charge

Lambda n'a pas de rôles liés au service, contrairement à Lambda@Edge. Pour plus d'informations, consultez la section Rôles liés aux services pour Lambda @Edge dans le manuel Amazon Developer Guide. CloudFront

Amazon Lightsail

Lightsail prend partiellement en charge les autorisations au niveau des ressources et ABAC. Pour de plus amples informations, veuillez consulter la rubrique Actions, ressources et clés de condition pour Amazon Lightsail.

Amazon Managed Streaming for Apache Kafka (MSK)

Vous pouvez associer une politique de cluster à un cluster Amazon MSK configuré pour une connectivité multi-VPC.

AWS Network Manager

AWS Le Cloud WAN prend également en charge les rôles liés aux services. Pour plus d'informations, consultez la section Rôles liés au service AWS Cloud WAN dans le guide Amazon VPC AWS Cloud WAN.

Amazon Relational Database Service

Amazon Aurora est un moteur de base de données relationnelle entièrement géré compatible avec MySQL et PostgreSQL. Vous pouvez choisir Aurora MySQL ou Aurora PostgreSQL en tant qu'option de moteur de base de données lorsque vous configurez de nouveaux serveurs de base de données via Amazon RDS. Pour plus d'informations, consultez Gestion des identités et des accès pour Amazon Aurora dans le Manuel de l'utilisateur Amazon Aurora.

Amazon Rekognition

Les politiques basées sur les ressources sont uniquement prises en charge pour la copie des modèles d'étiquettes personnalisées Amazon Rekognition.

AWS Resource Groups

Les utilisateurs peuvent endosser un rôle avec une politique qui autorise les opérations de groupes de ressources.

Amazon SageMaker

Les rôles liés aux services sont actuellement disponibles pour les postes de SageMaker studio et de SageMaker formation.

AWS Security Token Service

AWS STS ne dispose pas de « ressources », mais permet de restreindre l'accès de la même manière aux utilisateurs. Pour de plus amples informations, veuillez consulter Refus de l'accès aux informations d'identification de sécurité temporaires par nom.

Seules certaines des opérations d'API pour les appels d' AWS STS assistance avec des informations d'identification temporaires. Pour plus d'informations, consultez Comparaison de vos options d'API.

Amazon Simple Email Service

Vous pouvez uniquement utiliser les autorisations au niveau des ressources dans les instructions de politique qui font référence à des actions liées à l'envoi d'e-mails, comme `ses:SendEmail` ou `ses:SendRawEmail`. Pour obtenir des instructions de politique qui font référence à d'autres actions, l'élément Resource peut uniquement contenir `*`.

Seule l'API Amazon SES prend en charge les informations d'identification de sécurité temporaires. L'interface SMTP Amazon SES ne prend pas en charge les informations d'identification SMTP qui sont dérivées d'informations d'identification de sécurité temporaires.

Amazon Simple Storage Service

Amazon S3 prend en charge l'autorisation basée sur des balises pour les ressources d'objet uniquement.

Amazon S3 prend en charge les rôles liés au service pour Amazon S3 Storage Lens.

AWS Trusted Advisor

L'accès à l'API Trusted Advisor se fait via l' AWS Support API et est contrôlé par les politiques AWS Support IAM.

Amazon Virtual Private Cloud

Dans une politique utilisateur IAM, vous ne pouvez pas limiter les autorisations à un point de terminaison d'un VPC Amazon spécifique. Tout élément `Action` qui inclut les actions d'API

`ec2:*VpcEndpoint*` ou `ec2:DescribePrefixLists` doit spécifier « `"Resource": "*"` ». Pour de plus amples informations, veuillez consulter Gestion des identités et des accès pour les points de terminaison d'un VPC et les services de points de terminaison d'un VPC dans le Guide AWS PrivateLink .

Amazon VPC prend en charge l'attachement d'une seule politique de ressource à un point de terminaison d'un VPC afin de limiter les éléments accessibles via ce point de terminaison. Pour plus d'informations sur l'utilisation des politiques basées sur les ressources pour contrôler l'accès à celles-ci à partir de points de terminaison Amazon VPC spécifiques, consultez Control d'accès à l'utilisation des politiques de point de terminaison dans le Guide AWS PrivateLink .

Amazon VPC n'a pas de rôles liés à un service, mais il en a un. AWS Transit Gateway Pour plus d'informations, consultez la section Utiliser des rôles liés à un service pour la passerelle de transit dans le guide Amazon VPC. AWS Transit Gateway

AWS X-Ray

X-Ray ne prend pas en charge les autorisations au niveau des ressources pour toutes les actions.

X-Ray prend en charge le contrôle d'accès basé sur les balises pour les groupes et les règles d'échantillonnage.

Signature des demandes AWS d'API

> **Important**
>
> Si vous utilisez un AWS SDK (voir Exemples de code et bibliothèques) ou un outil de ligne de AWS commande (CLI) pour envoyer des demandes d'API AWS, vous pouvez ignorer cette section car les clients du SDK et de la CLI authentifient vos demandes à l'aide des clés d'accès que vous fournissez. À moins que vous n'ayez une bonne raison de ne pas le faire, nous vous recommandons de toujours utiliser un kit SDK ou la CLI.
> Dans les régions qui prennent en charge plusieurs versions de signature, les demandes de signature manuelle impliquent que vous devez spécifier la version de signature utilisée. Lorsque vous fournissez des demandes à des points d'accès multi-régions, les kits SDK et la CLI utiliseront automatiquement Signature version 4A sans configuration supplémentaire.

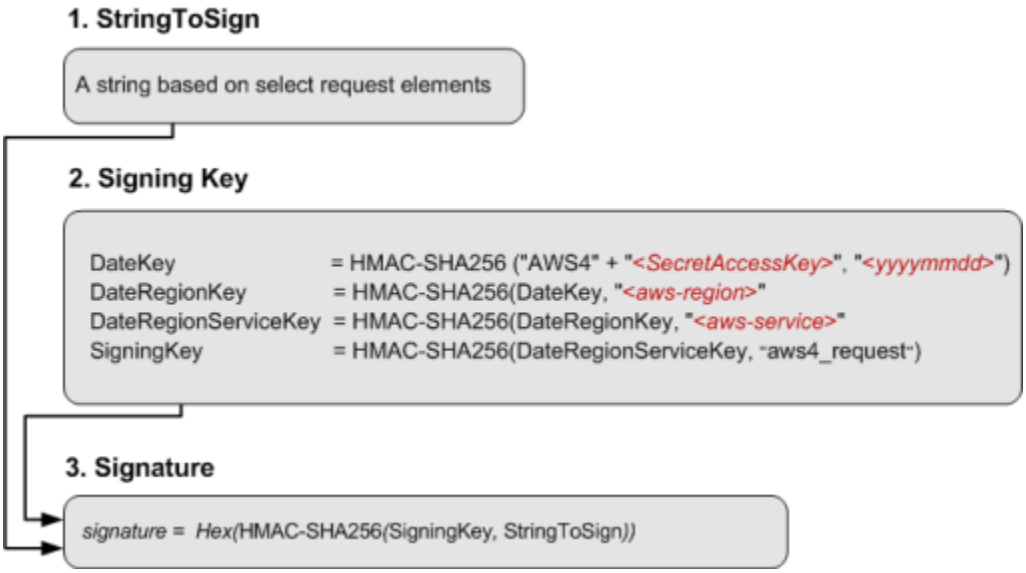
Les informations d'authentification que vous envoyez dans une demande doivent inclure une signature. Pour calculer une signature, vous devez d'abord concaténer certains éléments de

demande pour former une chaîne, appelée chaîne à signer. Vous utilisez ensuite une clé de signature pour calculer le code d'authentification de message utilisant hash (HMAC) de la chaîne à signer.

Dans AWS la version 4 de Signature, vous n'utilisez pas votre clé d'accès secrète pour signer la demande. À la place, vous utilisez d'abord votre clé d'accès secrète pour dériver une clé de signature. La clé de signature dérivée est spécifique à la date, au service et à la région. Pour plus d'informations sur la dérivation d'une clé de signature dans d'autres langages de programmation, consultez Demander des exemples de signature.

Signature La version 4 est le protocole AWS de signature. AWS prend également en charge une extension, Signature Version 4A, qui prend en charge les signatures pour les demandes d'API multirégionales. Pour plus d'informations, consultez le a-signing-examples projet sigv4 sur GitHub.

Le schéma suivant illustre le processus général de calcul d'une signature.



- La chaîne à signer dépend du type de demande. Par exemple, lorsque vous utilisez l'en-tête d'autorisation HTTP ou les paramètres de requête pour l'authentification, vous utilisez une combinaison variable d'éléments de demande pour créer la chaîne à signer. Pour une requête HTTP POST, la politique `POST` de la demande est la chaîne que vous signez.
- Pour clé de signature, le schéma montre une série de calculs, dont le résultat de chaque étape est intégré à l'étape suivante. La dernière étape est la clé de signature.
- Lorsqu'un AWS service reçoit une demande authentifiée, il recrée la signature à l'aide des informations d'authentification contenues dans la demande. Si les signatures correspondent, le service traite la demande. Sinon, il rejette la demande.

Table des matières

- Quand signer des demandes ?
- Pourquoi les demandes sont-elles signées ?
- Éléments d'une signature de demande d' AWS API
- Méthodes d'authentification
- Création d'une demande AWS d'API signée
- Demander des exemples de signature
- Résoudre les problèmes liés aux requêtes signées pour les API AWS

Quand signer des demandes ?

Lorsque vous écrivez du code personnalisé qui envoie des demandes d'API AWS, vous devez inclure le code qui signe les demandes. Vous pouvez écrire du code personnalisé pour les raisons suivantes :

- Vous utilisez un langage de programmation pour lequel il n'existe aucun kit SDK AWS .
- Vous avez besoin d'un contrôle total sur la manière dont les demandes sont envoyées AWS.

Pourquoi les demandes sont-elles signées ?

Le processus de signature aide à sécuriser les demandes de différentes façons :

- Vérifier l'identité du demandeur

  Les demandes authentifiées nécessitent une signature que vous créez à l'aide de vos clés d'accès (ID de clé d'accès, clé d'accès secrète). Si vous utilisez des informations d'identification de sécurité temporaires, les calculs de signature nécessitent également un jeton de sécurité. Pour plus d'informations, veuillez consulter la rubrique Accès par programmation des informations d'identification de sécuritéAWS.
- Protéger les données en transit

  Pour éviter qu'une demande ne soit falsifiée pendant son transit, certains de ses éléments sont utilisés pour calculer son hachage (digest) et la valeur de hachage obtenue est incluse comme partie intégrante de la demande. Lorsqu'un Service AWS reçoit la demande, il utilise les mêmes informations pour calculer un hachage et le compare à la valeur de hachage de votre demande. Si les valeurs ne correspondent pas, AWS refuse la demande.

- Assurer une protection contre les attaques potentielles par relecture

  Dans la plupart des cas, une demande doit parvenir AWS dans les cinq minutes suivant l'horodatage indiqué dans la demande. Dans le cas contraire, AWS refuse la demande.

Éléments d'une signature de demande d' AWS API

> **Important**
>
> À moins que vous n'utilisiez AWS les SDK ou la CLI, vous devez écrire du code pour calculer les signatures qui fournissent des informations d'authentification dans vos demandes. Le calcul des AWS signatures dans Signature Version 4 peut être une tâche complexe, et nous vous recommandons d'utiliser les AWS SDK ou la CLI dans la mesure du possible.

Chaque demande HTTP/HTTPS utilisant Signature Version 4 doit contenir ces éléments.

Eléments

- Spécification du point de terminaison
- Action
- Paramètres d'action
- Date
- Informations d'authentification

Spécification du point de terminaison

Spécifie le nom DNS du point de terminaison auquel vous envoyez la requête. Ce nom contient généralement le code du service et la région. Par exemple, le nom du point de terminaison Amazon DynamoDB de la région `us-east-1` est `dynamodb.us-east-1.amazonaws.com`.

Pour les requêtes HTTP/1.1, vous devez inclure l'en-tête `Host`. Pour les requêtes HTTP/2, vous pouvez utiliser l'en-tête `:authority` ou `Host`. Utilisez uniquement l'en-tête `:authority` à des fins de conformité avec la spécification HTTP/2. Tous les services ne prennent pas en charge les requêtes HTTP/2.

Pour les points de terminaison pris en charge par chaque service, consultez Points de terminaison et quotas de service dans Références générales AWS.

Action

Spécifie une action d'API pour le service. Par exemple, l'action DynamoDB `CreateTable` ou l'action Amazon EC2 `DescribeInstances`.

Pour les actions prises en charge par chaque service, consultez la référence d'autorisation des services.

Paramètres d'action

Spécifie les paramètres de l'action spécifiée dans la requête. Chaque action d' AWS API comporte un ensemble de paramètres obligatoires et facultatifs. La version de l'API est généralement un paramètre obligatoire.

Pour connaître les paramètres pris en charge par une action d'API, consultez la référence d'API du service.

Date

Spécifie la date et l'heure de la requête. L'inclusion de la date et de l'heure dans la requête permet d'empêcher des tiers d'intercepter cette dernière et de la soumettre à nouveau ultérieurement. La date que vous spécifiez dans l'étendue des informations d'identification doit correspondre à la date de votre requête.

L'horodatage doit être au format UTC et au format ISO 8601 suivant : AAAAMMJJTHHMMSSZ. Par exemple, `20220830T123600Z`. N'incluez pas de millisecondes dans l'horodatage.

Vous pouvez utiliser un en-tête `date` ou `x-amz-date`, ou inclure `x-amz-date` comme un paramètre de requête. Si nous ne trouvons pas d'en-tête `x-amz-date`, nous cherchons un en-tête `date`.

Informations d'authentification

Chaque demande que vous envoyez doit inclure les informations suivantes. AWS utilise ces informations pour garantir la validité et l'authenticité de la demande.

- Algorithme : `AWS 4-HMAC-SHA256` permet de spécifier la version 4 de la signature à l'aide de l'algorithme de hachage `HMAC-SHA256`.
- Identification : chaîne composée de votre identifiant de clé d'accès, de la date au format AAAAMMJJ, du code de région, du code de service et de la chaîne de terminaison

`aws4_request`, séparés par des barres obliques (/). Le code Région, le code de service et la chaîne de terminaison doivent utiliser des caractères minuscules.

```
AKIAIOSFODNN7EXAMPLE/YYYYMMDD/region/service/aws4_request
```

- En-têtes signés : en-têtes HTTP à inclure dans la signature, séparés par des points-virgules (;). Par exemple, `host;x-amz-date`.
- Signature : chaîne codée en hexadécimal qui représente la signature calculée. Vous devez calculer la signature en utilisant l'algorithme que vous avez spécifié dans le paramètre `Algorithm`.

Méthodes d'authentification

> **Important**
>
> À moins que vous n'utilisiez AWS les SDK ou la CLI, vous devez écrire du code pour calculer les signatures qui fournissent des informations d'authentification dans vos demandes. Le calcul des AWS signatures dans Signature Version 4 peut être une tâche complexe, et nous vous recommandons d'utiliser les AWS SDK ou la CLI dans la mesure du possible.

Vous pouvez exprimer les informations d'authentification à l'aide de l'une des méthodes suivantes.

En-tête d'autorisation HTTP

L'en-tête HTTP `Authorization` est l'une des méthodes d'authentification des demandes les plus courantes. Toutes les opérations de l'API REST (à l'exception des chargements via un navigateur utilisant des demandes POST) nécessitent cet en-tête. Pour plus d'informations sur la valeur de l'en-tête d'autorisation et sur le calcul de la signature et des options associées, consultez Authentification des demandes : utilisation de l'en-tête d'autorisation (AWS signature version 4) dans le manuel Amazon S3 API Reference.

Voici un exemple de valeur d'en-tête `Authorization`. Des sauts de ligne sont ajoutés à cet exemple pour faciliter la lecture. Dans votre code, l'en-tête doit être une chaîne continue. Il n'y a pas de virgule entre l'algorithme et les informations d'identification, mais les autres éléments doivent être séparés par des virgules.

```
Authorization: AWS 4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
```

```
SignedHeaders=host;range;x-amz-date,
Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024
```

Le tableau suivant décrit les différents composants de la valeur de l'en-tête d'autorisation dans l'exemple précédent :

Composant	Description
Autorisation	Algorithme utilisé pour calculer la signature. Vous devez fournir cette valeur lorsque vous utilisez la version 4 de AWS signature pour l'authentification. La chaîne indique la version de AWS signature 4 (`AWS 4`) et l'algorithme de signature (`HMAC-SHA256`).
Informations d'identification	Votre ID de clé d'accès et les informations relatives au champ d'application, notamment la date, la région et le service, qui ont été utilisés pour calculer la signature. Cette chaîne présente le format suivant : `<your-access-key-id>/<date>/ <aws-region>/<aws-service>/ aws4_request` Où : la valeur `<date>` est spécifiée à l'aide du format AAAAMMJJ. La valeur `<aws-service>` est s3 lors de l'envoi d'une demande à Amazon S3.
SignedHeaders	Une liste d'en-têtes de demande séparés par des points-virgules que vous avez utilisés pour calculer `Signature`. La liste inclut uniquement les noms d'en-tête, qui doivent être en minuscules. Par exemple : host ; range ; x-amz-date

Composant	Description
Signature	La signature de 256 bits exprimée en 64 caractères hexadécimaux minuscules. Exemples :`fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024` Veuillez noter que les calculs de signature varient en fonction de l'option que vous choisissez pour transférer la charge utile.

Paramètres des chaînes de requête

Vous pouvez utiliser une chaîne de requête pour exprimer entièrement une demande dans une URL. Dans ce cas, vous utilisez les paramètres de requête pour fournir les informations de demande, y compris les informations d'authentification. Comme la signature de la demande fait partie de l'URL, ce type d'URL est généralement appelé URL présignée. Vous pouvez utiliser des URL présignées pour intégrer des liens cliquables au format HTML, qui peuvent être valides jusqu'à sept jours. Pour plus d'informations, consultez Authentification des demandes : utilisation des paramètres de requête (AWS Signature version 4) dans le manuel Amazon S3 API Reference.

Voici un exemple d'URL présignée. Des sauts de ligne sont ajoutés à cet exemple pour faciliter la lecture :

```
https://s3.amazonaws.com/examplebucket/test.txt ?
X-Amz-Algorithm=AWS4-HMAC-SHA256 &
X-Amz-Credential=<your-access-key-id>/20130721/us-east-1/s3/aws4_request &
X-Amz-Date=20130721T201207Z &
X-Amz-Expires=86400 &
X-Amz-SignedHeaders=host &X-Amz-Signature=<signature-value>
```

> **Note**
>
> La valeur `X-Amz-Credential` dans l'URL affiche le caractère « / » uniquement pour des raisons de lisibilité. En pratique, il doit être encodé sous la forme %2F. Par exemple :

```
&X-Amz-Credential=<your-access-key-id>%2F20130721%2Fus-
east-1%2Fs3%2Faws4_request
```

Le tableau suivant décrit les paramètres de requête de l'URL qui fournissent des informations d'authentification.

Nom du paramètre de la chaîne d'interrogation	Description
X-Amz-Algorithm	Identifie la version de AWS Signature et l'algorithme que vous avez utilisé pour calculer la signature. Pour la version 4 de AWS Signature, vous définissez la valeur de ce paramètre sur. `AWS 4-HMAC-SHA256` Cette chaîne identifie AWS Signature Version 4 (AWS 4) et l'algorithme HMAC-SHA256 (`HMAC-SHA256`).
X-Amz-Credential	Outre l'ID de votre clé d'accès, ce paramètre indique également l'étendue (AWS région et service) pour laquelle la signature est valide. Cette valeur doit correspondre au champ d'application que vous utilisez dans les calculs de signature, comme indiqué dans la section suivante. Le format général de cette valeur de paramètre est le suivant : `<your-access-key-id>/<date>/<AWS Region>/<AWS-service>/aws4_request` Par exemple : `AKIAIOSFODNN7EXAMPLE/20130721/us-east-1/s3/aws4_request` Pour obtenir la liste des chaînes AWS régionales, consultez la section Points de terminaison

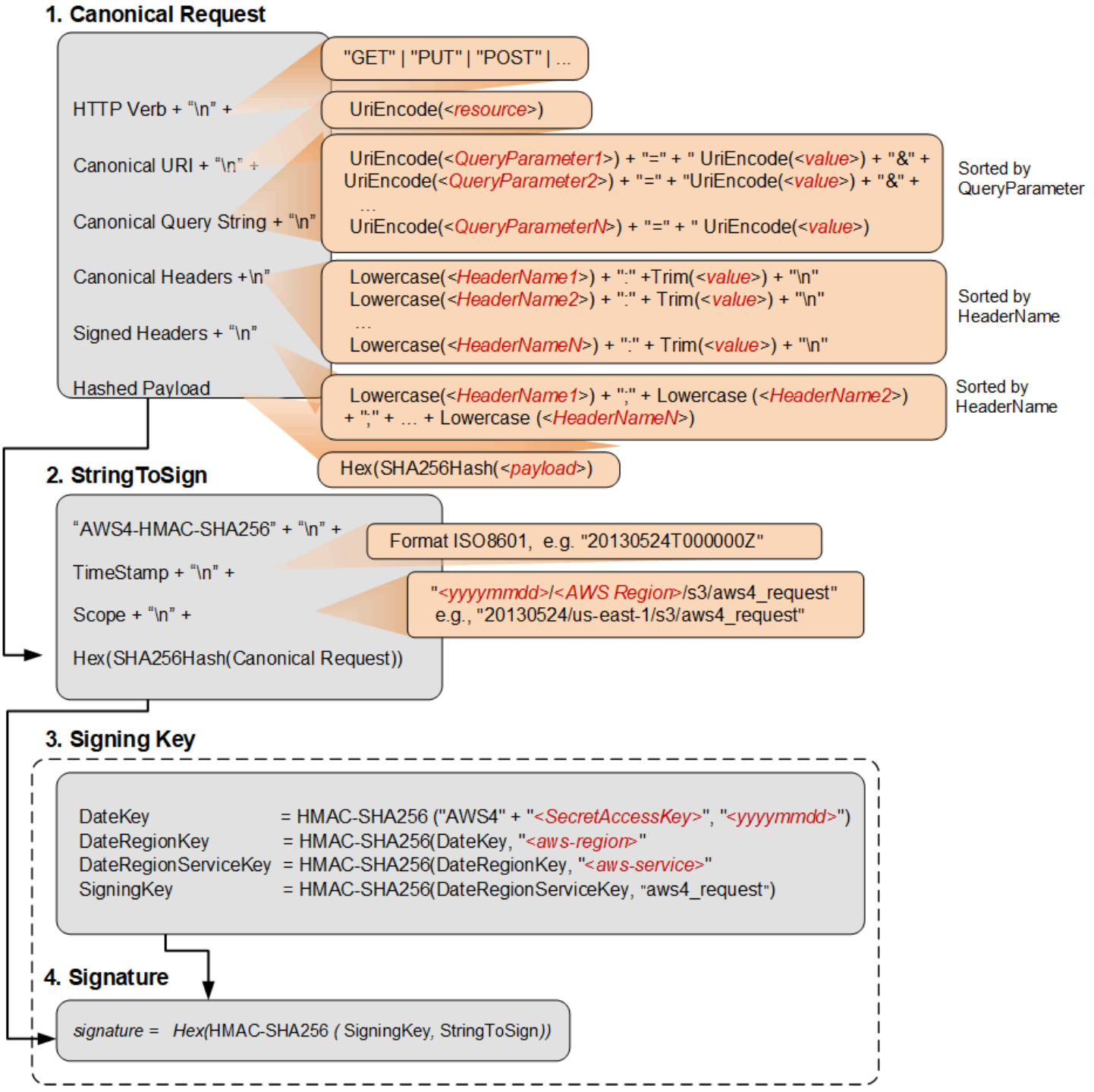
Nom du paramètre de la chaîne d'interrogation	Description
	régionaux dans le manuel de référence AWS général.
X-Amz-Date	Le format de date et d'heure doit respecter la norme ISO 8601 et doit être au format `yyyyMMddTHHmmssZ` . Par exemple, si la date et l'heure étaient « 01/08/2016 15:32:41.982-700 », elles doivent d'abord être converties en UTC (temps universel coordonné), puis soumises sous la forme « 20160801T223241Z ».
X-Amz-Expires	Indique la période, en secondes, pendant laquelle l'URL présignée générée est valide. Par exemple, 86 400 (24 heures). Cette valeur est un nombre entier. La valeur minimale que vous pouvez définir est 1 et la valeur maximale est de 604 800 (sept jours). Une URL présignée peut être valide pendant sept jours au maximum, car la clé de signature que vous utilisez pour le calcul de la signature est valide pendant sept jours au maximum.
X-Amz- SignedHeaders	Répertorie les en-têtes que vous avez utilisés pour calculer la signature. Les en-têtes suivants sont obligatoires pour les calculs de signature : • L'en-tête de l'hôte HTTP. • Tous les en-têtes x-amz-* que vous prévoyez d'ajouter à la demande. Pour plus de sécurité, vous devez signer tous les en-têtes de demande que vous prévoyez d'inclure dans votre demande.

Nom du paramètre de la chaîne d'interrogation	Description
X-Amz-Signature	Fournit la signature pour authentifier votre demande. Cette signature doit correspondre à la signature que le service calcule, sinon, le service refuse la demande. Par exemple, `733255ef022bec3f2a8701cd61d4b371f3f28c9f193a1f02279211d48d5193d7` Les calculs de signature sont décrits dans la section suivante.
X-Amz-Security-Token	Paramètre d'information d'identification facultatif si vous utilisez des informations d'identification provenant du service STS.

Création d'une demande AWS d'API signée

> **Important**
>
> Si vous utilisez un AWS SDK (voir Exemples de code et bibliothèques) ou un outil de ligne de AWS commande (CLI) pour envoyer des demandes d'API AWS, vous pouvez ignorer cette section car les clients du SDK et de la CLI authentifient vos demandes à l'aide des clés d'accès que vous fournissez. À moins que vous n'ayez une bonne raison de ne pas le faire, nous vous recommandons de toujours utiliser un kit SDK ou la CLI.
> Dans les régions qui prennent en charge plusieurs versions de signature, les demandes de signature manuelle impliquent que vous devez spécifier la version de signature utilisée. Lorsque vous fournissez des demandes à des points d'accès multi-régions, les kits SDK et la CLI utiliseront automatiquement Signature version 4A sans configuration supplémentaire.

Voici un aperçu du processus de création d'une requête signée. Pour calculer une signature, vous avez d'abord besoin d'une chaîne à signer. Vous calculez ensuite un hachage HMAC-SHA256 de la chaîne à signer à l'aide d'une clé de signature. Le schéma suivant illustre le processus, y compris les différents composants de la chaîne que vous créez pour la signature.



Le tableau suivant décrit les fonctions présentées dans le schéma. Vous devez implémenter du code pour ces fonctions. Pour plus d'informations, consultez les exemples de code dans les AWS SDK.

Fonction	Description
`Lowercase()`	Convertit la chaîne en minuscules.
`Hex()`	Encodage en minuscules en base 16.
`SHA256Hash()`	Fonction de hachage cryptographique Secure Hash Algorithm (SHA, algorithme de hachage sécurisé).
`HMAC-SHA256()`	Calcule HMAC à l'aide de l'algorithme SHA256 avec la clé de signature fournie. Il s'agit de la signature finale.
`Trim()`	Supprimez tous les espaces blancs de début ou de fin.
`UriEncode()`	L'URI code chaque octet. UriEncode() doit appliquer les règles suivantes : • L'URI encode tous les octets sauf les caractères non réservés : A-Z, a-z, 0-9, -, ., _ et ~. • Le caractère d'espace est un caractère réservé qui doit être codé sous la forme « %20 » (et non sous la forme « + »). • Chaque octet encodé par URI est formé par un « % » et la valeur hexadécimale à deux chiffres de l'octet. • Les lettres de la valeur hexadécimale doivent être en majuscules, par exemple « %1A ». • Encodez la barre oblique « / » partout sauf dans le nom de la clé de l'objet. Par exemple, si le nom de clé de l'objet est `photos/Jan/sample.jpg` , la barre oblique du nom de la clé n'est pas encodée.

Fonction	Description
	Important Les UriEncode fonctions standard fournies par votre plate-forme de développement peuvent ne pas fonctionner en raison de différenc es de mise en œuvre et de l'ambiguï té associée dans les RFC sous-jace ntes. Nous vous recommandons d'écrire votre propre UriEncode fonction personnalisée pour garantir le bon fonctionnement de votre encodage. Pour voir un exemple de UriEncode fonction en Java, consultez la section Utilitaires Java sur le GitHub site Web.

Note

Lorsque vous signez vos demandes, vous pouvez utiliser la version 4 de AWS signature ou AWS la version 4A de signature. La principale différence entre les deux est déterminée par la façon dont la signature est calculée. Avec AWS la version 4A de Signature, la signature n'inclut pas d'informations spécifiques à la région et est calculée à l'aide de l'`AWS 4-ECDSA-P256-SHA256`algorithme.

informations d’identification de sécurité temporaires

Au lieu d'utiliser des informations d'identification à long terme pour signer une demande, vous pouvez utiliser des informations de sécurité temporaires fournies par AWS Security Token Service (AWS STS).

Lorsque vous utilisez des informations d'identification de sécurité temporaires, vous devez ajouter`X-Amz-Security-Token` à l'en-tête Authorization ou à la chaîne de requête pour contenir le jeton de

session. Certains services nécessitent que vous ajoutiez `X-Amz-Security-Token` à la requête canonique. Pour les autres services, vous n'avez qu'à ajouter `X-Amz-Security-Token` à la fin, une fois que vous avez calculé la signature. Consultez la documentation de chacun Service AWS pour plus de détails.

Récapitulatif des étapes de signature

Étape 1 : créer une requête canonique

Organisez le contenu de votre demande (hôte, action, en-têtes, etc.) dans un format standard canonique. La demande canonique est l'une des entrées utilisées pour créer une chaîne de connexion. Pour plus de détails, consultez Éléments d'une signature de demande d' AWS API.

Étape 2 : créer un hachage de la requête canonique

Obtenez une clé de signature en effectuant une succession d'opérations de hachage par clé (opérations HMAC) à la date de la demande, dans la région et au service, avec votre clé d'accès AWS secrète comme clé pour l'opération de hachage initiale.

Étape 3 : créer une chaîne à signer

Créez une chaîne de connexion avec la demande canonique et des informations supplémentaires telles que l'algorithme, la date de la demande, la portée des informations d'identification et le digest (hachage) de la demande canonique.

Étape 4 : calculer la signature

Une fois la clé de signature dérivée, vous pouvez calculer la signature en exécutant une opération de hachage à clé sur la chaîne de connexion. Utilisez la clé de signature dérivée comme la clé de hachage pour cette opération.

Étape 5 : ajouter la signature à la requête

Une fois la signature calculée, ajoutez-la à un en-tête HTTP ou à la chaîne de requête de la demande.

Étape 1 : créer une requête canonique

Créez une requête canonique en concaténant les chaînes suivantes, séparées par des fins de ligne. Cela permet de garantir que la signature que vous calculez et la signature AWS calculée peuvent correspondre.

```
<HTTPMethod>\n
<CanonicalURI>\n
<CanonicalQueryString>\n
<CanonicalHeaders>\n
<SignedHeaders>\n
<HashedPayload>
```

- *HTTPMethod* : la méthode HTTP, telle que GET, PUT, HEAD et DELETE.
- *CanonicalUri*— La version codée en URI de l'URI du composant du chemin absolu, en commençant par le «/» qui suit le nom de domaine et jusqu'à la fin de la chaîne ou jusqu'au point d'interrogation (« ? ») si vous avez des paramètres de chaîne de requête. Si le chemin d'accès absolu est vide, utilisez un caractère barre oblique (/). L'URI de l'exemple suivant, `/examplebucket/myphoto.jpg`, est le chemin absolu et vous n'encodez pas la « / » dans le chemin absolu :

  ```
  http://s3.amazonaws.com/examplebucket/myphoto.jpg
  ```

- *CanonicalQueryString*— Les paramètres de chaîne de requête codés en URI. Vous encodez individuellement chaque nom et chaque valeur par URI. Vous devez également trier les paramètres de la chaîne de requête canonique par ordre alphabétique par nom de clé. Le tri s'effectue après l'encodage. La chaîne de requête dans l'exemple d'URI suivant est la suivante :

  ```
  http://s3.amazonaws.com/examplebucket?prefix=somePrefix&marker=someMarker&max-keys=2
  ```

  La chaîne de requête canonique est la suivante (des sauts de ligne sont ajoutés à cet exemple pour faciliter la lecture) :

  ```
  UriEncode("marker")+"="+UriEncode("someMarker")+"&"+
  UriEncode("max-keys")+"="+UriEncode("20") + "&" +
  UriEncode("prefix")+"="+UriEncode("somePrefix")
  ```

  Lorsqu'une requête cible une sous-ressource, la valeur du paramètre de requête correspondant est une chaîne vide (« »). Par exemple, l'URI suivant identifie la sous-ressource ACL sur le compartiment `examplebucket` :

  ```
  http://s3.amazonaws.com/examplebucket?acl
  ```

CanonicalQueryString Dans ce cas, c'est le suivant :

```
UriEncode("acl") + "=" + ""
```

Si l'URI ne contient pas de « ? » , la demande ne contient aucune chaîne de requête et vous définissez la chaîne de requête canonique sur une chaîne vide (« »). Vous devrez tout de même inclure le « \n ».

- *CanonicalHeaders*— Liste des en-têtes de requêtes avec leurs valeurs. Les paires de nom et de valeur d'en-tête individuelles sont séparées par le caractère de saut de ligne (« \n »). Voici un exemple de CanonicalHeader :

```
Lowercase(<HeaderName1>)+":"+Trim(<value>)+"\n"
Lowercase(<HeaderName2>)+":"+Trim(<value>)+"\n"
...
Lowercase(<HeaderNameN>)+":"+Trim(<value>)+"\n"
```

CanonicalHeaders la liste doit inclure les éléments suivants :

- En-tête `host` HTTP.
- Si l'`Content-Type`en-tête est présent dans la demande, vous devez l'ajouter à la *CanonicalHeaders*liste.
- Les en-têtes `x-amz-*` que vous prévoyez d'inclure dans votre demande doivent également être ajoutés. Par exemple, si vous utilisez des informations d'identification de sécurité temporaires, vous devez inclure `x-amz-security-token` dans votre demande. Vous devez ajouter cet en-tête dans la liste des *CanonicalHeaders*.

> **Note**
>
> L'`x-amz-content-sha256`en-tête est obligatoire pour les AWS requêtes Amazon S3. Il fournit un hachage de la charge utile de la requête. S'il n'y a pas de charge utile, vous devez fournir le hachage d'une chaîne vide.

Chaque nom d'en-tête doit :

- utiliser des caractères minuscules ;
- apparaître par ordre alphabétique ;

- être suivi de deux points (:).

Pour les valeurs, vous devez :

- supprimer toutes les espaces de début ou de fin ;
- convertir des espaces séquentielles en une espace unique ;
- séparez les valeurs d'un en-tête à valeurs multiples par des virgules.
- Vous devez inclure l'en-tête hôte (HTTP/1.1) ou l'en-tête :authority (HTTP/2), ainsi que tout en-tête `x-amz-*` dans la signature. Vous pouvez éventuellement inclure d'autres en-têtes standard dans la signature, tels que content-type.

Les fonctions `Lowercase()` et `Trim()` utilisées dans cet exemple sont décrites dans la section précédente.

L'exemple suivant est une chaîne `CanonicalHeaders`. Les noms d'en-tête sont en minuscules et sont triés.

```
host:s3.amazonaws.com
x-amz-content-sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
x-amz-date:20130708T220855Z
```

> **Note**
>
> Aux fins du calcul d'une signature d'autorisation, seuls l'hôte et les en-têtes `x-amz-*` sont obligatoires. Toutefois, afin d'éviter toute altération des données, vous devez envisager d'inclure tous les en-têtes dans le calcul de la signature.

- *`SignedHeaders`*— Une liste triée par ordre alphabétique et séparée par des points-virgules de noms d'en-tête de demande en minuscules. Les en-têtes de demande de la liste sont les mêmes que ceux que vous avez inclus dans la chaîne `CanonicalHeaders`. Par exemple, pour l'exemple précédent, la valeur de *`SignedHeaders`*serait la suivante :

```
host;x-amz-content-sha256;x-amz-date
```

- *HashedPayload*— Chaîne créée à l'aide de la charge utile contenue dans le corps de la requête HTTP comme entrée d'une fonction de hachage. Cette chaîne utilise des caractères hexadécimaux minuscules.

  ```
  Hex(SHA256Hash(<payload>)
  ```

  Si la demande ne contient aucune charge utile, vous calculez un hachage de la chaîne vide comme suit :

  ```
  Hex(SHA256Hash(""))
  ```

  Le hachage renvoie la valeur suivante :

  ```
  e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
  ```

  Par exemple, lorsque vous chargez un objet à l'aide d'une demande PUT, vous fournissez les données de l'objet dans le corps du message. Lorsque vous récupérez un objet à l'aide d'une demande GET, vous calculez le hachage de la chaîne vide.

Étape 2 : créer un hachage de la requête canonique

Créez un hachage (digest) de la requête canonique avec le même algorithme que celui que vous avez utilisé pour créer le hachage de la charge utile. Le hachage de la requête canonique est une chaîne de caractères hexadécimaux en minuscules.

Étape 3 : créer une chaîne de connexion

Créez une chaîne en concaténant les chaînes suivantes, séparées par des fins de ligne. Ne terminez pas cette chaîne par un caractère de saut de ligne.

```
Algorithm \n
RequestDateTime \n
CredentialScope  \n
HashedCanonicalRequest
```

- *Algorithme* : algorithme utilisé pour créer le hachage de la requête canonique. Pour SHA-256, l'algorithme est `AWS4-HMAC-SHA256`.

- *RequestDateTime*— Date et heure utilisées dans le champ d'application des informations d'identification. Cette valeur est l'heure UTC actuelle au format ISO 8601 (par exemple, 20130524T000000Z).
- *CredentialScope*— Le champ d'application des informations d'identification. Cela limite la signature résultante à la région et au service spécifiés. La chaîne a le format suivant : *AAAAMMJJ*/*region*/*service*/aws4_request.
- *HashedCanonicalRequest*— Le hachage de la demande canonique. Cette valeur est calculée à l'étape 2.

L'exemple suivant est une chaîne de connexion.

```
"AWS4-HMAC-SHA256" + "\n" +
timeStampISO8601Format + "\n" +
<Scope> + "\n" +
Hex(SHA256Hash(<CanonicalRequest>))
```

Étape 4 : calculer la signature

Dans AWS la version 4 de Signature, au lieu d'utiliser vos clés d' AWS accès pour signer une demande, vous créez une clé de signature limitée à une région et à un service spécifiques en tant qu'information d'authentification que vous ajouterez à votre demande.

```
DateKey = HMAC-SHA256("AWS4"+"<SecretAccessKey>", "<YYYYMMDD>")
DateRegionKey = HMAC-SHA256(<DateKey>, "<aws-region>")
DateRegionServiceKey = HMAC-SHA256(<DateRegionKey>, "<aws-service>")
SigningKey = HMAC-SHA256(<DateRegionServiceKey>, "aws4_request")
```

Pour obtenir la liste des chaînes régionales, veuillez consulter la rubrique Points de terminaison régionaux dans les Références générales AWS .

Pour chaque étape, appelez la fonction de hachage avec la clé et les données requises. Le résultat de chaque appel à la fonction de hachage devient l'entrée du prochain appel à la fonction de hachage.

Entrée obligatoire

- Chaîne, `Key`, contenant votre clé d'accès secrète
- Chaîne `Date` contenant la date utilisée dans l'étendue des informations d'identification, au format AAAAMMJJ

- Chaîne, `Region`, contenant le code Région (par exemple, `us-east-1`)
- Chaîne, `Service`, contenant le code de service (par exemple, `ec2`)
- Chaîne de connexion que vous avez créée à l'étape précédente.

Pour calculer la signature

1. Concaténez « AWS4 » et la clé d'accès secrète. Appelez la fonction de hachage avec la chaîne concaténée en tant que clé et la chaîne de date en tant que données.

```
kDate = hash("AWS4" + Key, Date)
```

2. Appelez la fonction de hachage en utilisant le résultat de l'appel précédent en tant que clé et la chaîne Région en tant que données.

```
kRegion = hash(kDate, Region)
```

3. Appelez la fonction de hachage en utilisant le résultat de l'appel précédent en tant que clé et la chaîne de service en tant que données.

```
kService = hash(kRegion, Service)
```

4. Appelez la fonction de hachage en utilisant le résultat de l'appel précédent en tant que clé et « aws4_request » en tant que données.

```
kSigning = hash(kService, "aws4_request")
```

5. Appelez la fonction de hachage en utilisant le résultat de l'appel précédent en tant que clé et la chaîne de signature en tant que données. Le résultat est la signature sous forme de valeur binaire.

```
signature = hash(kSigning, string-to-sign)
```

6. Convertissez la signature de la représentation binaire à la représentation hexadécimale, en caractères minuscules.

Étape 5 : ajouter la signature à la requête

Example Exemple : en-tête d'autorisation

L'exemple suivant montre un entête `Authorization` pour l'action `DescribeInstances`. Pour des raisons de lisibilité, cet exemple est formaté avec des sauts de ligne. Dans votre code, il doit s'agir d'une chaîne continue. Il n'y a pas de virgule entre l'algorithme et `Credential`. Toutefois, les autres éléments doivent être séparés par des virgules.

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request,
SignedHeaders=host;x-amz-date,
Signature=calculated-signature
```

Example Exemple : requête avec des paramètres d'authentification dans la chaîne de requête

L'exemple suivant présente une requête pour l'action `DescribeInstances` qui inclut les informations d'authentification. Pour des raisons de lisibilité, cet exemple est formaté avec des sauts de ligne et n’est pas code en URL. Dans votre code, la chaîne de requête doit être une chaîne continue codée en URL.

```
https://ec2.amazonaws.com/?
Action=DescribeInstances&
Version=2016-11-15&
X-Amz-Algorithm=AWS4-HMAC-SHA256&
X-Amz-Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request&
X-Amz-Date=20220830T123600Z&
X-Amz-SignedHeaders=host;x-amz-date&
X-Amz-Signature=calculated-signature
```

Code source dans les AWS SDK

Les AWS SDK incluent le code source permettant de signer GitHub les demandes AWS d'API. Pour obtenir des exemples de code, veuillez consulter Exemples de projets dans le référentiel AWS d'échantillons

- AWS SDK for .NET — AWS4Signer.cs
- AWS SDK for C++ — AWSAuthV4Signer.cpp
- AWS SDK for Go — v4.go

- AWS SDK for Java — BaseAws4Signer.java
- AWS SDK for JavaScript — v4.js
- AWS SDK for PHP — SignatureV4.php
- AWS SDK for Python (Boto) — signers.py
- AWS SDK for Ruby — signer.rb

Demander des exemples de signature

Les exemples suivants de demandes de AWS signature montrent comment utiliser SigV4 pour signer des demandes envoyées sans le AWS SDK ni l'outil de ligne de AWS commande.

Chargement d'Amazon S3 basé sur le navigateur avec HTTP POST

Authentification des demandes : chargements basés sur le navigateur décrit la signature et les informations pertinentes qu'Amazon S3 utilise pour calculer la signature lors de la réception de la demande.

Exemple : le téléchargement basé sur un navigateur à l'aide de HTTP POST (AWS Using Signature Version 4) fournit plus d'informations grâce à un exemple de politique POST et à un formulaire que vous pouvez utiliser pour télécharger un fichier. L'exemple de politique et les informations d'identification fictives vous montre le flux de travail ainsi que la signature et le hachage de la politique qui en résultent.

Demandes authentifiées VPC Lattice

Exemples pour les demandes authentifiées par Signature Version 4 (SigV4) fournit des exemples en Python et en Java qui vous montrent comment signer les demandes avec et sans intercepteurs personnalisés.

Utilisation de Signature Version 4 avec Amazon Translate

L'utilisation de Signature Version 4 avec Amazon Translate montre comment utiliser un programme Python pour ajouter des informations d'authentification aux demandes Amazon Translate. L'exemple exécute une demande POST, crée une structure JSON qui contient le texte à traduire dans le corps (charge utile) de la demande, et transmet les informations d'authentification dans un en-tête Authorization.

Utilisation de Signature Version 4 avec Neptune

Exemple : connexion à Neptune à l'aide de Python avec la signature Signature Version 4 montre comment envoyer des demandes signées à Neptune à l'aide de Python. Cet exemple inclut des variations pour l'utilisation d'une clé d'accès ou d'informations d'identification temporaires.

Signature des demandes HTTP à S3 Glacier

Exemple de calcul de signature pour l'API de streaming vous guide dans les détails de la création d'une signature pour Upload Archive (POST archive), l'une des deux API de streaming de S3 Glacier.

Envoi de demandes HTTP à Amazon SWF

Envoyez des demandes HTTP à Amazon SWF présente le contenu de l'en-tête d'une demande JSON envoyée à Amazon SWF.

Calcul de signature pour les API de streaming dans Amazon OpenSearch Service

La signature d'une demande de recherche Amazon OpenSearch Service avec le AWS SDK pour PHP version 3 inclut un exemple d'envoi de requêtes HTTP signées à OpenSearch Amazon Service.

Exemples de projets dans le référentiel AWS d'échantillons

Les exemples de projets suivants montrent comment signer des demandes pour envoyer des requêtes d'API Rest à AWS des services utilisant des langages courants tels que Python, Node.js, Java, C#, Go et Rust.

Projets de Signature Version 4a

Le projet sigv4-signing-examples fournit des exemples de la façon de signer des requêtes avec Sigv4a pour envoyer des requêtes d'API Rest avec des langages Services AWS courants tels que Python, Node.js, Java, C#, Go et Rust.

Le a-signing-examples projet sigv4 fournit des exemples de signature de demandes d'API multirégionales, par exemple des points d'accès multirégionaux dans Amazon S3.

Publier sur AWS IoT Core

Le code Python pour publier à AWS IoT Core l'aide du protocole HTTPS fournit des conseils sur la façon de publier des messages à AWS IoT Core l'aide du protocole HTTPS et de l'authentification AWS SigV4. Il possède deux implémentations de référence, l'une en Python et l'autre en NodeJs.

L'application .Net Framework pour publier à AWS IoT Core l'aide du protocole HTTPS fournit des conseils sur la façon de publier des messages à AWS IoT Core l'aide du protocole HTTPS et de l'authentification AWS SigV4. Ce projet inclut également une implémentation correspondant à .NET Core.

Résoudre les problèmes liés aux requêtes signées pour les API AWS

> **Important**
>
> À moins que vous n'utilisiez AWS les SDK ou la CLI, vous devez écrire du code pour calculer les signatures qui fournissent des informations d'authentification dans vos demandes. Le calcul de la signature SigV4 peut s'avérer complexe et nous vous recommandons d'utiliser les kits SDK ou la CLI AWS dans la mesure du possible.

Lorsque vous développez du code qui crée une demande signée, vous pouvez recevoir le protocole HTTP 403 `SignatureDoesNotMatch` de Services AWS. Ces erreurs signifient que la valeur de signature de votre requête HTTP AWS ne correspond pas à la signature Service AWS calculée. Les erreurs HTTP 401 `Unauthorized` sont renvoyées lorsque les autorisations ne permettent pas à l'appelant d'effectuer la requête.

Les demandes d'API peuvent renvoyer une erreur si :

- La demande d'API n'est pas signée et elle utilise l'authentification IAM.
- Les informations d'identification IAM utilisées pour signer la demande sont incorrectes ou ne sont pas autorisées à invoquer l'API.
- La signature de la demande d'API signée ne correspond pas à la signature calculée par le service AWS .
- L'en-tête de demande d'API est incorrect.

> **Note**
>
> Mettez à jour votre protocole de AWS signature de la version 2 (SigV2) à la version AWS Signature 4 (SigV4) avant d'explorer d'autres solutions aux erreurs. Les services, tels qu'Amazon S3, et les régions ne prennent plus en charge la signature SigV2.

Causes possibles :

- Erreurs d'informations d'identification
- Erreurs de requête canonique et de chaîne de signature
- Erreurs d’étendue des informations d'identification
- Erreurs de signature de clé

Erreurs d'informations d'identification

Assurez-vous que la demande d'API est signée avec SigV4. Si la demande d'API n'est pas signée, il se peut que vous receviez le message d'erreur : `Missing Authentication Token`. Ajoutez la signature manquante et renvoyez la demande.

Vérifiez que les informations d'identification de la clé d'accès et de la clé secrète sont correctes. Si la clé d'accès est incorrecte, le message d'erreur suivant peut s'afficher : `Unauthorized`. Assurez-vous que l'entité utilisée pour signer la demande est autorisée à effectuer cette dernière. Pour plus de détails, consultez Résolution des problèmes liés aux messages d'erreur d'accès rejeté.

Erreurs de requête canonique et de chaîne de signature

Si le calcul de la requête canonique est incorrect dans Étape 2 : créer un hachage de la requête canonique ou Étape 3 : créer une chaîne de connexion, l'étape de vérification de la signature effectuée par le service échoue avec le message d'erreur :

```
The request signature we calculated does not match the signature you provided
```

Lorsque le AWS service reçoit une demande signée, il recalcule la signature. S'il existe des différences entre les valeurs, les signatures ne correspondent pas. Comparez la requête canonique et la chaîne à votre demande signée avec la valeur du message d'erreur. Modifiez le processus de signature en cas de différence.

> **Note**
>
> Vous pouvez également vérifier que vous n'avez pas envoyé la requête via un proxy qui modifie les en-têtes ou la requête.

Example Exemple de requête canonique

```
GET                                                        -------- HTTP method
/                                                          -------- Path. For API stage
 endpoint, it should be /{stage-name}/{resource-path}
                                                           -------- Query string key-
value pair. Leave it blank if the request doesn't have a query string.
content-type:application/json                              -------- Header key-value
 pair. One header per line.
host:0123456789.execute-api.us-east-1.amazonaws.com        -------- Host and x-amz-date
 are required headers for all signed requests.
x-amz-date:20220806T024003Z

content-type;host;x-amz-date                               -------- A list of signed
 headers
d167e99c53f15b0c105101d468ae35a3dc9187839ca081095e340f3649a04501        -------- Hash
 of the payload
```

Pour vérifier que la clé secrète correspond à l'ID de la clé d'accès, vous pouvez les tester avec une implémentation fonctionnelle connue. Par exemple, utilisez un AWS SDK ou la AWS CLI pour envoyer une demande à AWS.

En-tête de la demande d'API

Assurez-vous que l'en-tête d'autorisation SigV4 que vous avez ajouté dans Étape 4 : calculer la signature contient la clé d'informations d'identification correcte similaire à la suivante :

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=example-generated-signature
```

Si la clé d'informations d'identification est manquante ou incorrecte, vous pouvez recevoir le message d'erreur suivant : `Authorization header requires 'Credential' parameter. Authorization header requires 'Signature' parameter.`. Assurez-vous que la demande d'autorisation SigV4 contient également la date de la demande en utilisant l'en-tête HTTP `Date` ou `x-amz-date`.

Erreurs d’étendue des informations d'identification

Le champ d'application des informations d'identification dans Étape 3 : créer une chaîne de connexion limite une signature à une date, une région et un service spécifiques. Cette chaîne présente le format suivant :

```
YYYYMMDD/region/service/aws4_request
```

> **Note**
>
> Si vous utilisez SigV4a, la région n'est pas incluse dans le champ d'application des informations d'identification.

Date

Si l'étendue des informations d'identification ne spécifie pas la même date que l'en-tête x-amz-date, l'étape de vérification de la signature échoue avec le message d'erreur suivant :

```
Date in Credential scope does not match YYYYMMDD from ISO-8601 version of date from
 HTTP
```

Si la requête indique une date future, l'étape de vérification de la signature échoue avec le message d'erreur suivant :

```
Signature not yet current: date is still later than date
```

Si la requête a expiré, l'étape de vérification de la signature échoue avec le message d'erreur suivant :

```
Signature expired: date is now earlier than date
```

Région

Si l'étendue des informations d'identification ne spécifie pas la même Région que la requête, l'étape de vérification de la signature échoue avec le message d'erreur suivant :

```
Credential should be scoped to a valid Region, not region-code
```

Service

Si l'étendue des informations d'identification ne spécifie pas le même service que l'en-tête host, l'étape de vérification de la signature échoue avec le message d'erreur suivant :

```
Credential should be scoped to correct service: 'service'
```

Chaîne de terminaison

Si l'étendue des informations d'identification ne se termine pas par aws4_request, l'étape de vérification de la signature échoue avec le message d'erreur suivant :

```
Credential should be scoped with a valid terminator: 'aws4_request'
```

Erreurs de signature de clé

Les erreurs causées par une dérivation incorrecte de la clé de signature ou une utilisation inappropriée du chiffrement sont plus difficiles à résoudre. Après avoir vérifié que la chaîne canonique et la chaîne à signer sont correctes, vous pouvez également vérifier l'existence de l'un des problèmes suivants :

- La clé d'accès secrète ne correspond pas à l'ID de clé d'accès que vous avez spécifiée.
- Votre code de dérivation de clé pose problème.

Pour vérifier que la clé secrète correspond à l'ID de la clé d'accès, vous pouvez les tester avec une implémentation fonctionnelle connue. Par exemple, utilisez un AWS SDK ou le AWS CLI pour faire une demande à AWS. Pour obtenir des exemples, consultez Demander des exemples de signature

Référence de politique JSON IAM

Cette section présente la syntaxe, les descriptions et des exemples détaillés des éléments, des variables et de la logique d'évaluation des politiques JSON dans IAM. Pour plus d'informations générales, consultez Présentation des politiques JSON.

Cette référence comprend les sections suivantes.

- Références des éléments de politique JSON IAM : en savoir plus sur les éléments que vous pouvez utiliser lorsque vous créez une politique. Consultez des exemples de politiques

supplémentaires, et découvrez les conditions et les types de données pris en charge, ainsi que la manière dont ils sont utilisés dans différents services.

- Logique d'évaluation de politiques— Cette section décrit les AWS demandes, comment elles sont authentifiées et comment AWS utilise les politiques pour déterminer l'accès aux ressources.
- Syntaxe du langage de politique JSON IAM – Cette section présente une syntaxe formelle du langage de stratégie utilisé pour créer des politiques dans IAM.
- AWS politiques gérées pour les fonctions professionnelles – Cette section répertorie toutes les politiques gérées par AWS qui correspondent directement à des fonctions professionnelles courantes du secteur de l'informatique. Utilisez ces politiques pour accorder les autorisations nécessaires à l'exécution des tâches prévues pour quelqu'un occupant une fonction professionnelle spécifique. Ces politiques consolident les autorisations pour de nombreux services en une politique unique.
- AWS clés contextuelles de condition globale— Cette section inclut une liste de toutes les clés de condition AWS globales que vous pouvez utiliser pour limiter les autorisations dans une politique IAM.
- Clés de contexte IAM et de AWS STS condition— Cette section inclut une liste de toutes les clés IAM et de AWS STS condition que vous pouvez utiliser pour limiter les autorisations dans une politique IAM.
- Actions, ressources et clés de condition pour les AWS services : cette section présente une liste de toutes les opérations d' AWS API que vous pouvez utiliser comme autorisations dans une politique IAM. Elle inclut également les clés de condition spécifiques au service qui peuvent être utilisées pour affiner davantage la demande.

Références des éléments de politique JSON IAM

Les documents de politique JSON se composent d'éléments. Les éléments sont répertoriés dans l'ordre général d'utilisation dans une politique. L'ordre des éléments n'a pas d'importance. Par exemple, l'élément `Resource` peut venir avant l'élément `Action`. Il n'est pas nécessaire de spécifier d'éléments `Condition` dans une politique. Pour en savoir plus sur la structure générale et la fonction d'un document de politique JSON, consultez Présentation des politiques JSON.

Certains éléments de politique JSON s'excluent mutuellement. Cela signifie que vous ne pouvez pas créer une politique qui les utilise ensemble. Par exemple, vous ne pouvez pas utiliser `Action` et `NotAction` dans la même instruction de politique. Les autres paires qui s'excluent mutuellement sont notamment `Principal`/`NotPrincipal` et `Resource`/`NotResource`.

Les détails inclus dans une politique varient pour chaque service, selon les actions autorisées par le service, les types de ressources qu'il contient, etc. Lorsque vous créez des politiques pour un service spécifique, il peut être utile de consulter des exemples de stratégie pour ce service. Pour consulter la liste de tous les services prenant en charge IAM et pour obtenir des liens vers la documentation de ces services relative à IAM et aux politiques, consultez AWS services qui fonctionnent avec IAM.

Lorsque vous créez ou modifiez une politique JSON, IAM peut effectuer une validation de politique pour vous aider à créer une politique efficace. IAM identifie les erreurs de syntaxe JSON, tandis que IAM Access Analyzer fournit des vérifications de politique supplémentaires avec des recommandations pour vous aider à affiner vos politiques. Pour en savoir plus sur la validation de politiques, veuillez consulter Validation de politiques IAM. Pour en savoir plus sur les vérifications des politiques IAM Access Analyzer et les recommandations exploitables, veuillez consulter Validation de politique IAM Access Analyzer.

Rubriques

- Éléments de politique JSON IAM : Version
- Éléments de politique JSON IAM : Id
- Éléments de politique JSON IAM : Statement
- Éléments de politique JSON IAM : Sid
- Éléments de politique JSON IAM : Effect
- AWS Éléments de politique JSON : Principal
- AWS Éléments de politique JSON : NotPrincipal
- Éléments de politique JSON IAM : Action
- Éléments de politique JSON IAM : NotAction
- Éléments de politique JSON IAM : Resource
- Éléments de politique JSON IAM : NotResource
- Éléments de politique JSON IAM : Condition
- Éléments des politiques IAM : variables et balises
- Éléments de politique JSON IAM : Types de données pris en charge

Éléments de politique JSON IAM : Version

> **Remarque pour lever l'ambiguïté**
>
> Cet élément de politique JSON `Version` ne représente pas la même chose qu'une version de politique. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. En revanche, une version de politique est créée lorsque vous apportez des modifications à une politique gérée par le client dans IAM. La politique modifiée ne remplace pas la politique existante. À la place, IAM crée une nouvelle version de la politique gérée. Si vous recherchez des informations sur les différentes versions prises en charge qui sont disponibles pour les politiques gérées, consultez the section called “Gestion des versions des politiques IAM”.

L'élément de politique `Version` spécifie les règles de syntaxe de langage qui doivent être utilisées pour traiter une politique. Pour utiliser toutes les fonctions de politique disponibles, incluez l'élément `Version` suivant à l'extérieur de l'élément `Statement` dans toutes vos politiques.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

IAM prend en charge les valeurs d'élément `Version` suivantes :

- `2012-10-17`. Il s'agit de la version actuelle du langage de politique ; vous devez toujours inclure un élément `Version` et définir sa valeur sur `2012-10-17`. Sinon, vous ne pouvez pas utiliser des fonctions telles que les variables de politique qui ont été introduites avec cette version.
- `2008-10-17`. Ceci est une version antérieure du langage de politique. Elle peut figurer dans des politiques existantes antérieures. N'utilisez pas cette version pour les nouvelles politiques ou lors de la mise à jour de politiques existantes. Les nouvelles fonctions, telles que les variables de politique, ne fonctionneront pas avec votre politique. Par exemple, les variables telles que

`${aws:username}` ne sont pas reconnues et sont traitées comme des chaînes littérales dans la politique.

Éléments de politique JSON IAM : Id

L'élément `Id` spécifie un identifiant facultatif pour la politique. L'ID est utilisé de différentes façons, selon les services. L'ID est autorisé dans les politiques basées sur les ressources, mais pas dans celles basées sur une identité.

Dans le cas de services vous permettant de définir un élément `ID`, il est recommandé d'utiliser un UUID (GUID) comme valeur, ou d'incorporer un UUID dans l'ID pour en garantir l'unicité.

```
{
  "Version": "2012-10-17",
  "Id": "cd3ad3d9-2776-4ef1-a904-4c229d1642ee",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Certains AWS services (par exemple, Amazon SQS ou Amazon SNS) peuvent nécessiter cet élément et avoir des exigences d'unicité pour celui-ci. Pour obtenir des informations spécifiques à chaque service sur la création de politiques, reportez-vous à la documentation du service que vous utilisez.

Éléments de politique JSON IAM : Statement

L'élément `Statement` constitue l'élément principal d'une politique. Cet élément est obligatoire. L'élément `Statement` peut contenir une seule instruction ou un tableau d'instructions. Chaque bloc d'instruction doit être placé entre accolades { }. Pour plusieurs instructions, le tableau doit être placé entre crochets [ ].

```
"Statement": [{...},{...},{...}]
```

L'exemple suivant illustre une politique qui contient un tableau de trois instructions dans un même élément `Statement`. (La politique vous permet d'accéder à votre propre « dossier principal » dans la console Amazon S3.) La politique inclut la variable `aws:username`, qui est remplacée par le nom utilisateur provenant de la demande lors de l'évaluation de la politique. Pour plus d'informations, voir Introduction.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::BUCKET-NAME",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}",
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}/*"
      ]
    }
  ]
}
```

Éléments de politique JSON IAM : Sid

Vous pouvez fournir un `Sid` (ID de déclaration) comme identifiant facultatif pour la déclaration de politique. Vous pouvez affecter une valeur `Sid` à chaque instruction d'un tableau de instructions. Vous pouvez utiliser la valeur `Sid` comme description de l'instruction de politique. Dans un service qui vous permet de spécifier un élément `ID` tel que SQS et SNS, la valeur `Sid` est un simple sous-identifiant de l'ID du document de politique. Dans IAM, la valeur `Sid` doit être unique dans une politique JSON.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExampleStatementID",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

L'élément `Sid` prend en charge les majuscules ASCII (A à Z), les minuscules (a à z) et les chiffres (0 à 9).

IAM n'expose pas le `Sid` dans l'API IAM. Il n'est pas possible d'extraire une instruction spécifique en vous basant sur cet ID.

> **Note**
>
> Certains AWS services (par exemple, Amazon SQS ou Amazon SNS) peuvent nécessiter cet élément et avoir des exigences d'unicité pour celui-ci. Pour obtenir des informations spécifiques à chaque service sur la création de politiques, reportez-vous à la documentation du service que vous utilisez.

Éléments de politique JSON IAM : Effect

L'élément `Effect` est requis ; il spécifie si l'instruction génère une autorisation ou un refus explicite. Les valeurs valides pour `Effect` sont `Allow` et `Deny`. La valeur `Effect` est sensible à la casse.

```
"Effect":"Allow"
```

Par défaut, l'accès aux ressources est refusé. Pour autoriser l'accès à une ressource, vous devez définir l'élément `Effect` sur `Allow`. Pour remplacer une autorisation (par exemple, pour remplacer une autorisation qui autrement est en vigueur), vous définissez l'élément `Effect` sur `Deny`. Pour plus d'informations, voir Logique d'évaluation de politiques.

AWS Éléments de politique JSON : Principal

Utilisez l'élément `Principal` dans une politique JSON basée sur les ressources pour spécifier le principal qui est autorisé ou non à accéder à une ressource.

Vous devez utiliser l'élément `Principal` dans les politiques basées sur les ressources. Plusieurs services prennent en charge les politiques basées sur les ressources, y compris IAM. Le type de politique basée sur les ressources IAM est une politique d'approbation de rôle. Dans les rôles IAM, utilisez l'élément `Principal` dans la politique d'approbation du rôle pour spécifier qui peut endosser le rôle. Lorsqu'il s'agit d'un accès entre comptes, vous devez spécifier l'identifiant à 12 chiffres du compte approuvé. Pour savoir si les principaux des comptes situés en dehors de votre zone de confiance (organisation ou compte de confiance) ont accès à vos rôles, consultez Qu'est-ce que l'Analyseur d'accès IAM ?.

> **Note**
>
> Après avoir créé le rôle, vous pouvez remplacer le compte par « * » pour autoriser tout le monde à endosser le rôle. Dans ce cas, nous vous recommandons vivement de limiter les personnes autorisées à accéder au rôle d'une autre manière, par exemple avec un élément `Condition` qui limite l'accès à certaines adresses IP. Votre rôle ne doit pas être accessible à n'importe qui !

D'autres exemples de ressources prenant en charge les politiques basées sur les ressources incluent un compartiment Amazon S3 ou une interface AWS KMS key.

Vous ne pouvez pas utiliser l'élément `Principal` dans une politique basée sur l'identité. Les politiques basées sur l'identité sont des politiques d'autorisations que vous attachez aux identités IAM (utilisateur, groupes ou rôles) . Dans ces cas, le principal est implicitement l'identité à laquelle est attachée la politique.

Rubriques

- Spécification d'un principal
- Compte AWS principes
- Principaux de rôles IAM
- Principaux de séance de rôle
- Principaux de l'utilisateur IAM
- Principaux d'IAM Identity Center
- AWS STS principes de session utilisateur fédérée
- AWS principes de service
- AWS principes de service dans les régions adhérentes
- Tous les principaux
- En savoir plus

Spécification d'un principal

Vous spécifiez un principal dans l'élément `Principal` d'une politique basée sur les ressources ou des clés de condition prenant en charge les principaux.

Vous pouvez spécifier l'un des principaux suivants dans une politique :

- Compte AWS et utilisateur root
- Rôles IAM
- Séances de rôle
- Utilisateurs IAM
- Séances d'utilisateur fédéré
- AWS services
- Tous les principaux

Vous ne pouvez pas identifier un groupe d'utilisateurs en tant que principal dans une politique (telle qu'une politique basée sur les ressources), car les groupes concernent les autorisations, et non l'authentification, et les principaux sont des entités IAM authentifiées.

Vous pouvez spécifier plus d'un principal pour chacun des types de principaux dans les sections suivantes, à l'aide d'un tableau. Les tableaux peuvent inclure une ou plusieurs valeurs. Lorsque vous

spécifiez plus d'un principal dans un élément, vous accordez des autorisations à chaque principal. Ceci est un OR logique et non un AND logique, car vous êtes authentifié comme un unique principal à la fois. Si vous incluez plus d'une valeur, utilisez des crochets ([et]) et délimitez chaque entrée du tableau par une virgule. L'exemple de politique suivant définit les autorisations pour le compte 123456789012 ou le compte 555555555555.

```
"Principal" : {
"AWS": [
  "123456789012",
  "555555555555"
  ]
}
```

> **Note**
>
> Vous ne pouvez pas utiliser un caractère générique pour faire correspondre une partie d'un nom de principal ou d'un ARN.

Compte AWS principes

Vous pouvez spécifier Compte AWS des identifiants dans l'`Principal`élément d'une politique basée sur les ressources ou dans des clés de condition qui prennent en charge les principaux. Cela délègue l'autorité sur le compte. Lorsque vous autorisez l'accès à un autre compte, un administrateur de ce compte doit alors accorder l'accès à une identité (utilisateur ou rôle IAM) de ce compte. Lorsque vous spécifiez un Compte AWS, vous pouvez utiliser l'ARN du compte (arn:aws:iam : : *account-ID:root*) ou une forme abrégée composée du préfixe suivi de l'identifiant du compte. `"AWS":`

Par exemple, soit un ID de compte `123456789012`, vous pouvez utiliser l'une des méthodes suivantes pour spécifier ce compte dans l'élément `Principal` :

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

```
"Principal": { "AWS": "123456789012" }
```

L'ARN du compte et l'ID de compte abrégé se comportent de la même manière. Les deux délèguent des autorisations au compte. L'utilisation de l'ARN du compte dans l'élément `Principal` ne limite pas les autorisations au seul utilisateur racine du compte.

> **Note**
>
> Lorsque vous enregistrez une politique basée sur les ressources qui inclut l'ID du compte abrégé, le service peut le convertir en ARN principal. Cela ne modifie pas la fonctionnalité de la politique.

Certains AWS services proposent des options supplémentaires pour spécifier un principal de compte. Par exemple, Amazon S3 vous permet de spécifier un ID d'utilisateur canonique à l'aide du format suivant :

```
"Principal": { "CanonicalUser":
 "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be" }
```

Vous pouvez également en spécifier plusieurs Compte AWS(ou ID utilisateur canonique) en tant que principal à l'aide d'un tableau. Par exemple, vous pouvez spécifier un principal dans une politique de compartiment à l'aide des trois méthodes.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::123456789012:root",
    "999999999999"
  ],
  "CanonicalUser": "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be"
}
```

Principaux de rôles IAM

Vous pouvez spécifier les ARN principaux du rôle IAM dans l'élément `Principal` d'une politique basée sur les ressources ou des clés de condition prenant en charge les principaux. Les rôles IAM sont des identités. Dans IAM, les identités sont des ressources auxquelles vous pouvez attribuer des autorisations. Les rôles font confiance à une autre identité authentifiée pour endosser ce rôle. Il s'agit notamment d'un principal dans l’interface AWS ou d'un utilisateur provenant d'un fournisseur d'identité externe (IdP). Lorsqu'un principal ou une identité endosse un rôle, ils reçoivent des informations d'identification de sécurité temporaires avec les autorisations du rôle endossé. Lorsqu'ils utilisent ces informations d'identification de session pour effectuer des opérations AWS, ils jouent le rôle principal de session.

Les rôles IAM sont des identités existantes dans IAM. Les rôles font confiance à une autre identité authentifiée, telle qu'une identité principale AWS ou un utilisateur provenant d'un fournisseur d'identité externe. Lorsqu'un principal ou une identité endossent un rôle, ils reçoivent des informations d'identification de sécurité temporaires. Ils peuvent ensuite utiliser ces informations d'identification comme principal de séance de rôle pour effectuer des opérations dans l'interface AWS.

Lorsque vous spécifiez un principal de rôle dans une politique basée sur les ressources, les autorisations effectives du principal sont limitées par tous les types de politique limitant les autorisations pour le rôle. Cela inclut les politiques de séance et les limites d'autorisations. Pour plus d'informations sur l'évaluation des autorisations effectives pour une séance de rôle, veuillez consulter Logique d'évaluation de politiques.

Pour spécifier l'ARN du rôle dans l'élément `Principal`, utilisez le format suivant :

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:role/role-name" }
```

> **Important**
>
> Si votre élément `Principal` dans une politique d'approbation de rôle contient un ARN qui pointe vers un rôle IAM spécifique, alors cet ARN se transforme en l'ID principal unique du rôle lorsque vous enregistrez la politique. Cela permet de réduire le risque d'escalade des privilèges par la suppression et la nouvelle création du rôle. Normalement, vous ne voyez pas cet ID dans la console, car IAM utilise une transformation inverse pour revenir au rôle ARN lorsque la politique d'approbation est affichée. Cependant, si vous supprimez le rôle, vous interrompez la relation. La politique ne s'applique plus, même si vous recréez le rôle, étant donné que le nouvel ID du principal du nouveau rôle ne correspond pas à l'ID stocké dans la politique d'approbation. Dans ce cas, l'ID principal apparaît dans les politiques basées sur les ressources, car il n'est plus AWS possible de le mapper à un ARN valide. Le résultat final est que si vous supprimez et recréez un rôle référencé dans l'élément `Principal` d'une politique d'approbation, vous devez modifier le rôle afin de remplacer l'ID du principal, qui est désormais incorrect, par l'ARN correct. L'ARN se transforme à nouveau en nouvel ID principal du rôle lorsque vous enregistrez la politique.

Vous pouvez également spécifier le principal du rôle comme principal dans une politique basée sur les ressources ou créer une politique d'autorisation étendue qui utilise la clé de condition `aws:PrincipalArn`. Lorsque vous utilisez cette clé, le principal de séance de rôle reçoit les

autorisations en fonction de l'ARN du rôle qui a été endossé, et non de l'ARN de la séance résultante. Comme il AWS ne convertit pas les ARN des clés de condition en identifiants, les autorisations accordées à l'ARN du rôle sont conservées si vous supprimez le rôle puis créez un nouveau rôle portant le même nom. Les types de politiques basées sur l'identité, tels que les limites de permissions ou les politiques de session, ne limitent pas les autorisations accordées à l'aide de la clé de condition `aws:PrincipalArn` avec un caractère générique(*) dans l'élément `Principal`, à moins que les politiques basées sur l'identité ne contiennent un rejet explicite.

Principaux de séance de rôle

Vous pouvez spécifier des séances de rôle dans l'élément `Principal`d'une politique basée sur les ressources ou des clés de condition prenant en charge les principaux. Lorsqu'un principal ou une identité endosse un rôle, ils reçoivent des informations d'identification de sécurité temporaires avec les autorisations du rôle endossé. Lorsqu'ils utilisent ces informations d'identification de session pour effectuer des opérations AWS, ils jouent le rôle principal de session.

Le format que vous utilisez pour un rôle principal de session dépend de l' AWS STS opération utilisée pour assumer le rôle.

En outre, les administrateurs peuvent concevoir un processus pour contrôler la façon dont les séances de rôle sont émises. Par exemple, ils peuvent fournir une solution en un clic à leurs utilisateurs qui crée un nom de séance prévisible. Si votre administrateur procède ainsi, vous pouvez utiliser des principaux de séance de rôle dans vos politiques ou clés de condition. Sinon, vous pouvez spécifier l'ARN du rôle comme principal dans la `aws:PrincipalArn` clé de condition. La façon dont vous spécifiez le rôle comme principal peut modifier les autorisations effectives pour la séance résultante. Pour plus d'informations, veuillez consulter Principaux de rôles IAM.

Principaux de session à rôle endossé

Un principal de session assumé est un principal de session qui résulte de l'utilisation de l'opération. AWS STS `AssumeRole` Pour de plus amples informations sur les principaux pouvant endosser un rôle à l'aide de cette opération, veuillez consulter Comparaison des opérations AWS STS d'API.

Pour spécifier l'ARN de la séance à rôle endossé dans l'élément `Principal`, utilisez le format suivant :

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:assumed-role/role-name/role-session-name" }
```

Lorsque vous spécifiez une session à rôle endossé dans un élément `Principal`, vous ne pouvez pas utiliser un caractère générique « * » pour signifier toutes les sessions. Les principaux doivent toujours nommer une session spécifique.

Principes des sessions de l'OIDC

Un principal de session OIDC est un principal de session qui résulte de l'utilisation de l' AWS STS `AssumeRoleWithWebIdentity`opération. Vous pouvez utiliser un fournisseur OIDC (IdP) externe pour vous connecter, puis assumer un rôle IAM à l'aide de cette opération. Cela tire parti de la fédération d'identité et génère une séance de rôle. Pour de plus amples informations sur les principaux pouvant endosser un rôle à l'aide de cette opération, veuillez consulter Comparaison des opérations AWS STS d'API.

Lorsque vous attribuez un rôle à un fournisseur OIDC, vous obtenez ce type spécial de principal de session qui inclut des informations sur le fournisseur OIDC.

Utilisez ce type de principal dans votre politique pour autoriser ou rejeter l'accès en fonction du fournisseur d'identité web approuvé. Pour spécifier l'ARN de session de rôle OIDC dans l'`Principal`élément d'une politique de confiance des rôles, utilisez le format suivant :

```
"Principal": { "Federated": "cognito-identity.amazonaws.com" }
```

```
"Principal": { "Federated": "www.amazon.com" }
```

```
"Principal": { "Federated": "graph.facebook.com" }
```

```
"Principal": { "Federated": "accounts.google.com" }
```

Principaux de séance SAML

Un principal de session SAML est un principal de session qui résulte de l'utilisation de l' AWS STS `AssumeRoleWithSAML`opération. Vous pouvez utiliser un fournisseur d'identité SAML externe (IdP) pour vous connecter, puis endosser un rôle IAM à l'aide de cette opération. Cela tire parti de la fédération d'identité et génère une séance de rôle. Pour de plus amples informations sur les principaux pouvant endosser un rôle à l'aide de cette opération, veuillez consulter Comparaison des opérations AWS STS d'API.

Lorsque vous émettez un rôle à partir d'un fournisseur d'identité SAML, vous obtenez ce type spécial de principal de séance qui comprend des informations sur le fournisseur d'identité SAML.

Utilisez ce type de principal dans votre politique pour autoriser ou rejeter l'accès en fonction du fournisseur d'identité SAML approuvé. Pour spécifier l'ARN de séance du rôle d'identité SAML dans l'élément `Principal` d'une politique d'approbation de rôle, utilisez le format suivant :

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:saml-provider/provider-name" }
```

Principaux de l'utilisateur IAM

Vous pouvez spécifier des utilisateurs IAM dans l'élément `Principal` d'une politique basée sur les ressources ou dans les clés de condition qui prennent en charge les principes.

> **Note**
>
> Dans un élément `Principal`, le nom utilisateur faisant partie d'Amazon Resource Name (ARN) est sensible à la casse.

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:user/user-name" }
```

```
"Principal": {
  "AWS": [
    "arn:aws:iam::AWS-account-ID:user/user-name-1",
    "arn:aws:iam::AWS-account-ID:user/user-name-2"
  ]
}
```

Lors de la spécification d'utilisateurs dans un élément `Principal`, il n'est pas possible d'utiliser un caractère générique (*) signifiant « tous les utilisateurs ». Les principaux doivent toujours nommer des utilisateurs spécifiques.

> **Important**
>
> Si votre élément `Principal` dans une politique d'approbation de rôle comporte un ARN qui pointe vers un utilisateur IAM spécifique, IAM transforme l'ARN en ID du principal unique de l'utilisateur lorsque vous enregistrez la politique. Cela permet de réduire le risque d'escalade des privilèges par la suppression et la nouvelle création de l'utilisateur. Cet ID n'est pas fréquent dans la console, car il existe également une transformation inverse, pour revenir à l'ARN de l'utilisateur, lorsque la politique d'approbation est affichée. Cependant, si vous

supprimez l'utilisateur, vous interrompez la relation La politique ne s'applique plus, même si vous recréez l'utilisateur. Cela est dû au fait que le nouvel ID du principal du nouvel utilisateur ne correspond pas à l'ID stocké dans la politique d'approbation. Dans ce cas, l'ID principal apparaît dans les politiques basées sur les ressources, car il n'est plus AWS possible de le mapper à un ARN valide. Par conséquent, si vous supprimez et recréez un utilisateur référencé dans l'élément `Principal` d'une politique d'approbation, vous devez modifier le rôle afin de remplacer l'ID du principal qui est désormais incorrect par l'ARN correct. IAM transforme à nouveau l'ARN en le nouvel ID du principal de l'utilisateur lorsque vous enregistrez la politique.

Principaux d'IAM Identity Center

Dans IAM Identity Center, le principal d'une politique basée sur les ressources doit être défini comme étant le principal Compte AWS . Pour spécifier l'accès, faites référence à l'ARN du rôle du jeu d'autorisations dans le bloc de conditions. Pour en savoir, consultez Référencement des jeux d'autorisations dans les politiques de ressources, Amazon EKS et AWS KMS (français non garanti) dans le Guide de l'utilisateur IAM Identity Center.

AWS STS principes de session utilisateur fédérée

Vous pouvez spécifier des séances d'utilisateur fédéré dans l'élément `Principal` d'une politique basée sur les ressources ou dans les clés de condition qui prennent en charge les principaux.

> **Important**
>
> AWS recommande d'utiliser des sessions utilisateur AWS STS fédérées uniquement lorsque cela est nécessaire, par exemple lorsqu'un accès utilisateur root est requis. Au lieu de cela, utilisez les rôles pour déléguer les autorisations.

Un principal de session utilisateur AWS STS fédéré est un principal de session qui résulte de l'utilisation de l' AWS STS `GetFederationToken`opération. Dans ce cas, AWS STS utilise la fédération d'identité comme méthode permettant d'obtenir des jetons d'accès temporaires au lieu d'utiliser les rôles IAM.

Dans AWS, les utilisateurs IAM ou an Utilisateur racine d'un compte AWS peuvent s'authentifier à l'aide de clés d'accès à long terme. Pour plus d'informations sur les principaux qui peuvent se fédérer à l'aide de cette opération, veuillez consulter Comparaison des opérations AWS STS d'API.

- Utilisateur fédéré IAM – Un utilisateur IAM se fédère à l'aide de l'opération `GetFederationToken` qui donne lieu à un principal de séance d'utilisateur fédéré pour cet utilisateur IAM.
- Utilisateur racine fédéré – Un utilisateur racine se fédère à l'aide de l'opération `GetFederationToken` qui donne lieu à un principal de séance d'utilisateur fédéré pour cet utilisateur racine.

Lorsqu'un utilisateur IAM ou un utilisateur root demande des informations d'identification temporaires pour AWS STS utiliser cette opération, il ouvre une session utilisateur fédérée temporaire. L'ARN de cette séance est basé sur l'identité originale qui a été fédérée.

Pour spécifier l'ARN de la séance d'utilisateur fédéré dans l'élément `Principal`, utilisez le format suivant :

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:federated-user/user-name" }
```

AWS principes de service

Vous pouvez spécifier AWS des services dans l'`Principal`élément d'une politique basée sur les ressources ou dans des clés de condition qui prennent en charge les principaux. Un principal de service est un identifiant pour un service.

Les rôles IAM qui peuvent être assumés par un AWS service sont appelés rôles de service. Les rôles de service doivent inclure une politique d'approbation. Les politiques d'approbation sont des politiques basées sur les ressources attachées à un rôle qui définit les principaux qui peuvent endosser le rôle. Certains rôles de service disposent de politiques d'approbation prédéfinies. Toutefois, dans certains cas, vous devez spécifier le principal du service dans la politique d'approbation. Le principal de service d'une politique IAM ne peut pas l'être`"Service": "*"`.

L'identifiant d'un principal de service inclut le nom du service et se présente généralement dans le format suivant :

service-name`.amazonaws.com`

Le principal de service est défini par le service. Vous pouvez rechercher le principal de service pour certains services en ouvrant l'interface AWS services qui fonctionnent avec IAM, en vérifiant si le service indique Yes (Oui) dans la colonne Service-linked role (rôle lié à un service) et en ouvrant le lien Yes (Oui) pour afficher la documentation du rôle lié à un service pour ce service. Recherchez la section Autorisations de rôles liés à un service correspondant à ce service pour afficher le principal du service.

L'exemple suivant illustre une politique qui est peut être attachée à un rôle de service. La politique permet à deux services, Amazon ECS et Elastic Load Balancing, d'endosser le rôle. Les services peuvent ensuite effectuer toutes les tâches autorisées par la politique d'autorisation affectée au rôle (non illustré). Pour définir plusieurs principaux de service, vous ne spécifiez pas deux éléments `Service` ; vous ne devez en avoir qu'un seul. À la place, vous utilisez un tableau de plusieurs principaux de service comme valeur d'un élément `Service` unique.

```
"Principal": {
    "Service": [
        "ecs.amazonaws.com",
        "elasticloadbalancing.amazonaws.com"
   ]
}
```

AWS principes de service dans les régions adhérentes

Vous pouvez lancer des ressources dans plusieurs AWS régions et vous devez vous inscrire dans certaines d'entre elles. Pour obtenir la liste complète des régions auxquelles vous devez adhérer, consultez la section Gestion des AWS régions dans le Références générales AWSguide.

Lorsqu'un AWS service d'une région optionnelle fait une demande dans la même région, le format du nom principal du service est identifié comme étant la version non régionalisée de son nom principal de service :

service-name`.amazonaws.com`

Lorsqu'un AWS service d'une région optionnelle adresse une demande interrégionale à une autre région, le format du nom principal du service est identifié comme étant la version régionalisée de son nom principal de service :

service-name`.`*{region}*`.amazonaws.com`

Par exemple, vous avez une rubrique Amazon SNS dans la région `ap-southeast-1` et un compartiment Amazon S3 dans la région d'adhésion `ap-east-1`. Vous voulez configurer les notifications du compartiment S3 pour publier des messages dans la rubrique SNS. Pour permettre au service S3 de publier des messages dans la rubrique SNS, vous devez accorder au principal de service S3 une autorisation `sns:Publish` via la stratégie d'accès basée sur les ressources de la rubrique.

Si vous spécifiez la version non régionalisée du principal de service S3, `s3.amazonaws.com`, dans la stratégie d'accès à la rubrique, la demande `sns:Publish` du compartiment vers la rubrique

échouera. L'exemple suivant indique le principal de service S3 non régionalisé dans l'élément de stratégie `Principal` de la stratégie d'accès à la rubrique SNS.

```
"Principal": { "Service": "s3.amazonaws.com" }
```

Étant donné que le compartiment se trouve dans une région d'adhésion et que la demande est faite en dehors de cette même région, le principal de service S3 apparaît comme le nom du principal de service régionalisé, `s3.ap-east-1.amazonaws.com`. Vous devez utiliser le nom principal du service régionalisé lorsqu'un AWS service d'une région optionnelle adresse une demande à une autre région. Une fois que vous avez spécifié le nom du principal de service régionalisé, si le compartiment adresse une demande `sns:Publish` à la rubrique SNS située dans une autre région, la demande sera réussie. L'exemple suivant indique le principal de service S3 régionalisé dans l'élément de stratégie `Principal` de la stratégie d'accès à la rubrique SNS.

```
"Principal": { "Service": "s3.ap-east-1.amazonaws.com" }
```

Les stratégies de ressources ou les listes d'autorisation basées sur le principal de service pour les demandes inter-région d'une région d'adhésion vers une autre région ne seront réussies que si vous spécifiez le nom du principal de service régionalisé.

> **Note**
>
> Pour les stratégies d'approbation de rôle IAM, nous recommandons d'utiliser le nom du principal de service non régionalisé. Les ressources IAM sont globales et le même rôle peut donc être utilisé dans n'importe quelle région.

Tous les principaux

Vous pouvez utiliser un caractère générique (*) pour spécifier tous les principaux dans l'élément `Principal` d'une politique basée sur les ressources ou dans les clés de condition prenant en charge les principaux. Politiques basées sur les ressources accorde des autorisations et les clés de condition sont utilisées pour limiter les conditions d'une déclaration de politique.

> **Important**
>
> Nous vous recommandons fortement de ne pas utiliser de caractère générique (*) dans l'élément `Principal` d'une politique basée sur les ressources avec un effet `Allow`, sauf si vous avez l'intention d'accorder un accès public ou anonyme. Sinon, indiquez les principaux,

les services ou les comptes AWS visés dans l'élément `Principal`, puis restreignez davantage l'accès dans l'élément `Condition`. Cela est particulièrement vrai pour les politiques de confiance des rôles IAM, car elles permettent à d'autres principaux de devenir un principal dans votre compte.

Pour les politiques basées sur les ressources, utiliser un caractère générique (*) avec un effet `Allow` accorde l'accès à tous les utilisateurs, y compris les utilisateurs anonymes (accès public). Pour les utilisateurs IAM et les principaux de rôle de votre compte, aucune autre autorisation n'est requise. Pour les principaux d'autres comptes, ils doivent également disposer d'autorisations basées sur l'identité dans leur compte qui les autorise à accéder à votre ressource. C'est ce que l'on appelle l'accès entre comptes.

Pour les utilisateurs anonymes, les éléments suivants sont équivalents :

```
"Principal": "*"
```

```
"Principal" : { "AWS" : "*" }
```

Vous ne pouvez pas utiliser un caractère générique pour faire correspondre une partie d'un nom de principal ou d'un ARN.

L'exemple suivant montre une politique basée sur les ressources qui peut être utilisée à la place de Spécifier l'élément `NotPrincipal` avec `Deny` dans le but de refuser explicitement tous les principaux, à l'exception de ceux qui sont spécifiés dans l'élément `Condition`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "UsePrincipalArnInsteadOfNotPrincipalWithDeny",
      "Effect": "Deny",
      "Action": "s3:*",
      "Principal": "*",
      "Resource": [
        "arn:aws:s3:::BUCKETNAME/*",
        "arn:aws:s3:::BUCKETNAME"
      ],
      "Condition": {
        "ArnNotEquals": {
```

```
          "aws:PrincipalArn": "arn:aws:iam::444455556666:user/user-name"
        }
      }
    }
  ]
}
```

En savoir plus

Pour plus d'informations, consultez les ressources suivantes :

- Exemples de politique de compartiment dans le Guide de l'utilisateur service de stockage simple Amazon
- Exemples de politiques pour Amazon SNS dans le Guide du développeur Amazon Simple Notification Service
- Exemples de politique Amazon SQS dans le Guide du développeur Amazon Simple Queue Service
- Politiques de clé dans le Guide du développeur AWS Key Management Service
- Identifiants de compte (français non garanti) dans Références générales AWS
- Fédération OIDC

AWS Éléments de politique JSON : NotPrincipal

Vous pouvez utiliser l'`NotPrincipal`élément pour refuser l'accès à tous les principaux, à l'exception de l'utilisateur IAM, de l'utilisateur fédéré, du rôle IAM Compte AWS, du AWS service ou de tout autre principal spécifié dans l'élément. `NotPrincipal`

Vous pouvez l'utiliser dans les politiques basées sur les ressources pour certains AWS services, notamment les points de terminaison VPC. Les politiques basées sur les ressources sont des politiques que vous intégrez directement à une ressource. Vous ne pouvez pas utiliser l'élément `NotPrincipal` dans une politique IAM basée sur l'identité ni dans une politique d'approbation du rôle IAM.

L'élément `NotPrincipal` doit être utilisé avec `"Effect":"Deny"`. Son utilisation avec `"Effect":"Allow"` n'est pas prise en charge.

> **Important**
>
> Très peu de scénarios requièrent l'utilisation de l'élément `NotPrincipal`. Nous vous recommandons d'envisager d'autres options d'autorisation avant de choisir `NotPrincipal`.

Lorsque vous utilisez l'élément `NotPrincipal`, il peut être difficile de résoudre les problèmes liés à plusieurs types de politique. Nous recommandons plutôt l'utilisation de la clé de contexte `aws:PrincipalArn` avec les opérateurs de condition ARN. Pour plus d'informations, consultez Tous les principaux.

Spécifier l'élément **`NotPrincipal`** avec **`Deny`**

Lorsque vous utilisez `NotPrincipal` avec `Deny`, vous devez également spécifier l'ARN du compte du principal non refusé. Dans le cas contraire, la politique peut refuser l'accès à l'ensemble du compte contenant le principal. En fonction du service que vous incluez dans votre politique, AWS peut valider d'abord le compte, puis l'utilisateur. Si un utilisateur assumant un rôle (une personne utilisant un rôle) est en cours d'évaluation, AWS vous pouvez d'abord valider le compte, puis le rôle, puis l'utilisateur assumé. Ce dernier est identifié par le nom de session de rôle qui est spécifié lorsqu'il endosse le rôle. Par conséquent, nous vous recommandons vivement d'inclure explicitement l'ARN du compte d'un utilisateur, ou inclure à la fois l'ARN d'un rôle et l'ARN du compte contenant le rôle.

Important

N'utilisez pas de déclarations de politique basée sur les ressources qui incluent un élément de politique `NotPrincipal` ayant un effet `Deny` sur les utilisateurs IAM ou les rôles auxquels est attachée une politique de limite des autorisations. L'élément `NotPrincipal` ayant un effet `Deny` refusera toujours tout principal IAM auquel est attachée une politique de limite des autorisations, quelles que soient les valeurs spécifiées dans l'élément `NotPrincipal`. Certains utilisateurs ou rôles IAM qui auraient autrement eu accès à la ressource n'y ont donc plus accès. Nous vous recommandons de modifier vos déclarations de politique basée sur les ressources afin d'utiliser l'opérateur de condition `ArnNotEquals` avec la clé de contexte `aws:PrincipalArn` dans le but de limiter l'accès plutôt que l'élément `NotPrincipal`. Pour plus d'informations sur les limites des autorisations, veuillez consulter Limites d'autorisations pour les entités IAM.

Note

En tant que bonne pratique, vous devez inclure les ARN pour le compte dans votre politique. Certains services nécessitent l'ARN du compte, bien que ce ne soit pas obligatoire dans tous les cas. Toutes les politiques existantes sans l'ARN requis continuent de fonctionner,

mais les nouvelles politiques qui incluent ces services doivent répondre à cette exigence. IAM n'assure pas le suivi de ces services ; par conséquent nous recommandons de toujours inclure le compte ARN.

Les exemples suivants montrent comment utiliser `NotPrincipal` et `"Effect": "Deny"` de manière optimale dans la même instruction de politique.

Example Exemple d'utilisateur IAM dans le même compte ou un compte différent

Dans l'exemple suivant, tous les principaux, à l'exception de l'utilisateur nommé Bob dans Compte AWS 444455556666, se voient explicitement refuser l'accès à une ressource. Notez qu'il est recommandé que l'`NotPrincipal`élément contienne à la fois l'ARN de l'utilisateur Bob et Compte AWS celui auquel Bob appartient (`arn:aws:iam::444455556666:root`). Si l'`NotPrincipal`élément ne contient que l'ARN de Bob, la politique peut avoir pour effet de refuser explicitement l'accès à Compte AWS celui qui contient l'utilisateur Bob. Dans certains cas, un utilisateur ne peut pas disposer de plus d'autorisations que son compte parent. Ainsi, si l'accès est refusé explicitement au compte de Bob, Bob serait également dans l'impossibilité d'accéder à la ressource.

Cet exemple fonctionne comme prévu lorsqu'il fait partie d'une déclaration de politique d'une stratégie basée sur les ressources attachée à une ressource identique ou différente Compte AWS (et non 444455556666). Cet exemple seul n'accorde pas d'accès à Bob ; il omet simplement Bob de la liste des principaux auxquels l'accès est explicitement refusé. Pour permettre à Bob d'accéder à la ressource, une autre instruction de politique doit explicitement accorder l'accès à l'aide de `"Effect": "Allow"`.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "NotPrincipal": {"AWS": [
            "arn:aws:iam::444455556666:user/Bob",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "s3:*",
        "Resource": [
            "arn:aws:s3:::BUCKETNAME",
            "arn:aws:s3:::BUCKETNAME/*"
        ]
```

```
    }]
}
```

Example Exemple de rôle IAM dans le même compte ou un compte différent

Dans l'exemple suivant, tous les principaux, à l'exception de l'utilisateur supposé nommé cross-account-audit-app dans Compte AWS 444455556666, se voient explicitement refuser l'accès à une ressource. Il est recommandé que l'`NotPrincipal`élément contienne l'ARN du rôle assumé user (cross-account-audit-app), du rôle (cross-account-read-only-role) et du rôle auquel appartient le rôle (Compte AWS 444455556666). Si l'élément `NotPrincipal` ne contient pas l'ARN du rôle, la politique refuserait explicitement l'accès au rôle. De la même façon, si l'élément `NotPrincipal` ne contient pas l'ARN de l' Compte AWS auquel appartient le rôle, la politique refuserait explicitement l'accès à l' Compte AWS ainsi qu'à toutes les entités qu'il contient. Dans certains cas, les utilisateurs assumant un rôle ne peuvent pas avoir plus d'autorisations que leur rôle parent, et les rôles ne peuvent pas avoir plus d'autorisations que leur parent. Ainsi Compte AWS, lorsque l'accès au rôle ou au compte est explicitement refusé, l'utilisateur du rôle assumé peut ne pas être en mesure d'accéder à la ressource.

Cet exemple fonctionne comme prévu lorsqu'il fait partie d'une déclaration de politique d'une stratégie basée sur les ressources attachée à une ressource d'une autre stratégie Compte AWS (et non 444455556666). Cet exemple en lui-même n'autorise pas l'accès à l'utilisateur assumé cross-account-audit-app, il ne figure que dans la liste des principaux utilisateurs explicitement refusés. cross-account-audit-app Pour donner cross-account-audit-app accès à la ressource, une autre déclaration de politique doit explicitement autoriser l'accès en utilisant`"Effect": "Allow"`.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "NotPrincipal": {"AWS": [
            "arn:aws:sts::444455556666:assumed-role/cross-account-read-only-role/cross-
account-audit-app",
            "arn:aws:iam::444455556666:role/cross-account-read-only-role",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "s3:*",
        "Resource": [
            "arn:aws:s3:::Bucket_AccountAudit",
            "arn:aws:s3:::Bucket_AccountAudit/*"
        ]
```

```
    }]
}
```

Lorsque vous spécifiez une session endossed-role dans un élément `NotPrincipal`, vous ne pouvez pas utiliser un caractère générique (*) pour signifier « toutes les sessions ». Les principaux doivent toujours nommer une session spécifique.

Éléments de politique JSON IAM : Action

L'élément `Action` décrit les actions spécifiques qui seront accordées ou refusées. Les instructions doivent inclure un élément `Action` ou `NotAction`. Chaque AWS service possède son propre ensemble d'actions qui décrivent les tâches que vous pouvez effectuer avec ce service. Par exemple, la liste des actions pour Amazon S3 se trouve dans Spécifier les autorisations dans une politique du guide de l'utilisateur d'Amazon Simple Storage Service, la liste des actions pour Amazon EC2 se trouve dans le manuel Amazon EC2 API Reference, et la liste des actions AWS Identity and Access Management pour se trouve dans le IAM API Reference. Pour consulter la liste d'actions d'autres services, reportez-vous à la documentation de référence de l'API pour le service requis.

Vous spécifiez une valeur en utilisant un espace de noms du service comme préfixe d'action (`iam`, `ec2` `sqs`, `sns`, `s3`, etc.), suivi du nom de l'action à autoriser ou refuser. Le nom doit correspondre à une action prise en charge par le service. Le préfixe et le nom d'action ne sont pas sensibles à la casse. Par exemple, `iam:ListAccessKeys` est identique à `IAM:listaccesskeys`. Les exemples suivants illustrent les éléments `Action` pour différents services.

Action Amazon SQS

```
"Action": "sqs:SendMessage"
```

Action Amazon EC2

```
"Action": "ec2:StartInstances"
```

Action IAM

```
"Action": "iam:ChangePassword"
```

Actions Amazon S3

```
"Action": "s3:GetObject"
```

Vous pouvez spécifier plusieurs valeurs pour l'élément `Action`.

```
"Action": [ "sqs:SendMessage", "sqs:ReceiveMessage", "ec2:StartInstances",
 "iam:ChangePassword", "s3:GetObject" ]
```

Vous pouvez utiliser un caractère générique (*) pour donner accès à toutes les actions proposées par le AWS produit concerné. Par exemple, l'élément `Action` suivant s'applique à toutes les actions S3.

```
"Action": "s3:*"
```

Vous pouvez aussi utiliser des caractères génériques (*) dans le nom d'action. Par exemple, l'élément `Action` suivant s'applique à toutes les actions IAM qui incluent la chaîne `AccessKey`, y compris `CreateAccessKey`, `DeleteAccessKey`, `ListAccessKeys` et `UpdateAccessKey`.

```
"Action": "iam:*AccessKey*"
```

Certains services vous permettent de limiter les actions disponibles. Par exemple, Amazon SQS vous permet de ne mettre à disposition qu'un sous-ensemble de toutes les actions Amazon SQS possibles. Dans ce cas, le caractère générique `*` ne permet pas de contrôler entièrement la file d'attente ; il autorise uniquement le sous-ensemble d'actions que vous avez partagées. Pour de plus amples informations, veuillez consulter Comprendre les autorisations dans le Guide du développeur Amazon Simple Storage Service.

Éléments de politique JSON IAM : NotAction

`NotAction` est un élément de politique avancé qui correspond de manière explicite à tout sauf à la liste spécifiée des actions. L'utilisation de `NotAction` peut entraîner une politique plus courte en répertoriant uniquement quelques actions qui ne devraient pas correspondre, plutôt que d'inclure une longue liste d'actions qui correspondront. Les actions spécifiées dans ne `NotAction` sont pas affectées par l'`Deny`effet `Allow` ou l'effet d'une déclaration de politique. En revanche, cela signifie que toutes les actions ou services applicables non répertoriés sont autorisés si vous utilisez l'effet `Allow`. En outre, ces actions ou services non répertoriés sont refusées si vous utilisez l'effet `Deny`. Lorsque vous utilisez `NotAction` avec l'élément `Resource`, vous fournissez la portée de la politique. C'est ainsi que l'on AWS détermine quelles actions ou quels services sont applicables. Pour plus d'informations, consultez l'exemple de politique suivant :

NotAction avec Allow

Vous pouvez utiliser l'`NotAction`élément dans une instruction `"Effect": "Allow"` pour donner accès à toutes les actions d'un AWS service, à l'exception des actions spécifiées dans`NotAction`. Vous pouvez l'utiliser avec l'élément `Resource` pour fournir la portée de la politique, en limitant les actions autorisées aux actions pouvant être exécutées sur la ressource spécifiée.

L'exemple suivant permet aux utilisateurs d'accéder à toutes les actions Amazon S3 pouvant être exécutées sur une ressource S3 sauf pour supprimer un compartiment. Il ne permet pas aux utilisateurs d'utiliser l'opération d'API S3 `ListAllMyBuckets`, car cette action nécessite la ressource « * ». Cette politique n'autorise pas non plus les actions dans d'autres services, car les actions d'autres services ne sont pas applicables aux ressources S3.

```
"Effect": "Allow",
"NotAction": "s3:DeleteBucket",
"Resource": "arn:aws:s3:::*",
```

Il peut arriver que vous souhaitiez autoriser l'accès à un grand nombre d'actions. En utilisant l'élément `NotAction`, vous inversez de manière efficace l'instruction, ce qui réduit la liste des actions. Par exemple, en raison AWS du grand nombre de services, vous souhaiterez peut-être créer une politique permettant à l'utilisateur de tout faire sauf d'accéder aux actions IAM.

L'exemple suivant permet aux utilisateurs d'accéder à toutes les actions de tous les AWS services, à l'exception d'IAM.

```
"Effect": "Allow",
"NotAction": "iam:*",
"Resource": "*"
```

Soyez vigilant lorsque vous utilisez l'élément `NotAction` et `"Effect": "Allow"` dans la même instruction ou une instruction différente dans une politique. `NotAction` correspond à tous les services et les actions qui ne sont pas répertoriés de manière explicite ou applicables à la ressource spécifiée, et peut accorder aux utilisateurs plus d'autorisations que vous n'auriez souhaité.

NotAction avec Deny

Vous pouvez utiliser l'élément `NotAction` dans une instruction avec `"Effect": "Deny"` pour refuser un accès à l'ensemble des ressources répertoriées sauf pour les actions spécifiées dans l'élément `NotAction`. Cette combinaison n'autorise pas les éléments répertoriés, mais à la place elle refuse de manière explicite les actions non répertoriées. Vous devez toujours autoriser les actions que vous souhaitez autoriser.

L'exemple conditionnel suivant refuse l'accès aux actions non-IAM si l'utilisateur n'est pas connecté lorsqu'il utilise MFA. Si l'utilisateur est connecté lorsqu'il utilise MFA, le test `"Condition"` échoue et l'instruction `"Deny"` finale n'a aucun effet. Notez, toutefois, que ceci n'accorde à l'utilisateur l'accès à aucune action, mais ne fait que refuser explicitement toutes les autres actions sauf les actions IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "DenyAllUsersNotUsingMFA",
        "Effect": "Deny",
        "NotAction": "iam:*",
        "Resource": "*",
        "Condition": {"BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}}
    }]
}
```

Pour obtenir un exemple de politique qui refuse l'accès aux actions en dehors de régions spécifiques, à l'exception des actions provenant de services spécifiques, veuillez consulter AWS: refuse l'accès AWS en fonction de la région demandée.

Éléments de politique JSON IAM : Resource

L'élément `Resource` spécifie les objets couverts par l'instruction. Les instructions doivent inclure un élément `Resource` ou `NotResource`. Vous spécifiez une ressource à l'aide d'un ARN. Pour de plus amples informations sur le format des ARN, veuillez consulter ARN IAM.

Chaque service dispose de son propre ensemble de ressources. Bien que vous utilisiez toujours un ARN pour spécifier une ressource, le contenu de l'ARN d'une ressource dépend du service et de la ressource. Pour plus d'informations sur la façon de spécifier une ressource, reportez-vous à la documentation du service pour lequel vous souhaitez rédiger une instruction.

> **Note**
>
> Certains services ne vous permettent pas de spécifier des actions pour des ressources individuelles ; toutes les actions répertoriées dans l'élément `Action` ou `NotAction` s'appliquent alors à l'ensemble des ressources du service. Dans ce cas, vous utilisez le caractère générique `*` dans l'élément `Resource`.

L'exemple suivant fait référence à une file d'attente Amazon SQS spécifique.

```
"Resource": "arn:aws:sqs:us-east-2:account-ID-without-hyphens:queue1"
```

L'exemple suivant fait référence à un utilisateur IAM nommé Bob dans un Compte AWS.

> **Note**
>
> Dans l'élément `Resource`, le nom de l'utilisateur IAM est sensible à la casse.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/Bob"
```

Utilisation de caractères génériques dans les ARN de ressources

Il est possible d'utiliser des caractères génériques dans l'ARN d'une ressource. Vous pouvez utiliser des caractères génériques (* et ?) dans des segments ARN (les parties séparées par des deux points) pour représenter n'importe quelle combinaison de caractères avec un astérisque (*) et n'importe quel caractère unique avec un point d'interrogation (?). Vous pouvez utiliser plusieurs * ou ? dans chaque segment. Si le caractère générique (*) est le dernier caractère d'un segment d'ARN de ressource, il peut s'étendre pour correspondre au-delà des limites des deux points. Nous vous recommandons d'utiliser des caractères génériques (* et ?) dans les segments d'ARN séparés par deux points.

> **Note**
>
> Vous ne pouvez pas utiliser de caractère générique dans le segment de service identifiant le AWS produit. Pour plus d'informations sur les segments ARN, consultez Amazon Resource Names (ARN)

L'exemple suivant fait référence à tous les utilisateurs IAM dont le chemin d'accès est `/accounting`.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/accounting/*"
```

L'exemple suivant fait référence à tous les éléments d'un compartiment Amazon S3 spécifique.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
```

Le caractère astérisque (*) peut se développer pour remplacer tout ce qui se trouve dans un segment, y compris des caractères tels qu'une barre oblique (/) qui pourrait autrement sembler être un délimiteur dans un espace de noms de service donné. Par exemple, considérez l'ARN Amazon S3 suivant comme la même logique d'extension générique qui s'applique à tous les services.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*/test/*"
```

Les caractères génériques de l'ARN s'appliquent à tous les objets suivants dans le compartiment, et pas seulement au premier objet répertorié.

```
DOC-EXAMPLE-BUCKET/1/test/object.jpg
DOC-EXAMPLE-BUCKET/1/2/test/object.jpg
DOC-EXAMPLE-BUCKET/1/2/test/3/object.jpg
DOC-EXAMPLE-BUCKET/1/2/3/test/4/object.jpg
DOC-EXAMPLE-BUCKET/1///test///object.jpg
DOC-EXAMPLE-BUCKET/1/test/.jpg
DOC-EXAMPLE-BUCKET//test/object.jpg
DOC-EXAMPLE-BUCKET/1/test/
```

Considérez les deux derniers objets de la liste précédente. Un nom d'objet Amazon S3 peut valablement commencer ou se terminer par la barre oblique (/) du délimiteur classique. Bien que « / » fonctionne comme un délimiteur, ce caractère n'a pas de signification spécifique lorsqu'il est utilisé dans un ARN de ressource. Il est traité de la même manière que n'importe quel autre caractère valide. L'ARN ne correspondrait pas aux objets suivants :

```
DOC-EXAMPLE-BUCKET/1-test/object.jpg
DOC-EXAMPLE-BUCKET/test/object.jpg
DOC-EXAMPLE-BUCKET/1/2/test.jpg
```

Spécification de plusieurs ressources

Vous pouvez spécifier plusieurs ressources. L'exemple suivant fait référence à deux tables DynamoDB.

```
"Resource": [
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/books_table",
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/magazines_table"
]
```

Utilisation de variables de politique dans les ARN de ressources

Dans l'élément `Resource`, vous pouvez utiliser des variables de politique JSON dans la partie de l'ARN qui identifie la ressource spécifique (autrement dit, dans la partie finale de l'ARN). Par exemple, vous pouvez utilisez la clé `{aws:username}` au sein de l'ARN de la ressource pour indiquer que le nom de l'utilisateur actuel doit être inclus dans le nom de la ressource. L'exemple suivant montre comment utiliser la clé `{aws:username}` dans un élément `Resource`. La politique autorise l'accès à une table Amazon DynamoDB correspondant au nom de l'utilisateur actuel.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:account-id:table/${aws:username}"
  }
}
```

Pour plus d'informations sur les variables de politique JSON, consultez Éléments des politiques IAM : variables et balises.

Éléments de politique JSON IAM : NotResource

`NotResource` est un élément de politique avancé qui correspond explicitement à chaque ressource, à l'exception de celles spécifiées. L'utilisation de `NotResource` peut entraîner une politique plus courte en répertoriant uniquement quelques ressources qui ne devraient pas correspondre, plutôt que d'inclure une longue liste de ressources qui correspondront. Ceci est particulièrement utile pour les politiques applicables au sein d'un même service AWS .

Prenons l'exemple d'un groupe nommé `HRPayroll`. Les membres de `HRPayroll` ne doivent pas être autorisés à accéder aux ressources Amazon S3 sauf au dossier `Payroll` dans le compartiment `HRBucket`. La politique suivante refuse de manière explicite l'accès à toutes les ressources Amazon S3 autres que celles répertoriées. Notez, toutefois, que cette politique n'accorde à l'utilisateur aucun accès aux ressources.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "s3:*",
```

```
    "NotResource": [
      "arn:aws:s3:::HRBucket/Payroll",
      "arn:aws:s3:::HRBucket/Payroll/*"
    ]
  }
}
```

En règle générale, pour refuser de manière explicite l'accès à une ressource vous rédigez une politique qui utilise `"Effect":"Deny"` et inclut un élément `Resource` qui répertorie individuellement chaque dossier. Toutefois, dans ce cas, chaque fois que vous ajoutez un dossier à `HRBucket`, ou une ressource à Amazon S3 à laquelle personne ne doit accéder, vous devez ajouter son nom à la liste dans `Resource`. Si vous utilisez un élément `NotResource` à la place, les utilisateurs verront leur accès aux nouveaux dossiers automatiquement refusé sauf si vous ajoutez les noms des dossiers à l'élément `NotResource`.

Lorsque vous utilisez `NotResource`, n'oubliez pas que les ressources spécifiées dans cet élément sont les seules ressources qui ne sont pas limitées. Toutes les ressources qui s'appliqueraient à l'action sont limitées. Dans l'exemple ci-dessus, la politique affecte uniquement les actions Amazon S3, et donc uniquement les ressources Amazon S3. Si l'action comprenait également des actions Amazon EC2, la politique ne refuserait pas l'accès aux ressources EC2. Pour savoir quelles actions d'un service permettent de spécifier l'ARN d'une ressource, consultez Actions, ressources et clés de condition pour les AWS services.

NotResource avec d'autres éléments

Vous ne devez jamais utiliser les éléments `"Effect": "Allow"`, `"Action": "*"` et `"NotResource": "arn:aws:s3:::HRBucket"` ensemble. Cette déclaration est très dangereuse, car elle autorise toutes les actions AWS sur toutes les ressources à l'exception du compartiment `HRBucket` S3. Elle autoriserait même l'utilisateur à s'ajouter une politique qui lui permettrait d'accéder à `HRBucket`. Ne le faites pas.

Soyez vigilant lorsque vous utilisez l'élément `NotResource` et `"Effect": "Allow"` dans la même instruction ou une instruction différente dans une politique. `NotResource` autorise tous les services et ressources qui ne sont pas répertoriés de manière explicite et peut accorder aux utilisateurs plus d'autorisations que vous n'auriez souhaité. L'utilisation de l'élément `NotResource` et `"Effect": "Deny"` dans la même instruction refuse les services et les ressources qui ne sont pas répertoriés de manière explicite.

Éléments de politique JSON IAM : Condition

L'élément `Condition` (ou bloc `Condition`) vous permet de spécifier des conditions lorsqu'une politique est appliquée. L'élément `Condition` est facultatif. Dans l'élément `Condition`, vous créez des expressions dans lesquelles vous utilisez des opérateurs de condition (égal, inférieur à, etc.) pour faire correspondre les clés et valeurs de contexte de la politique avec les clés et valeurs du contexte de la demande. Pour de plus amples informations sur le contexte de la demande, veuillez consulter Demande.

```
"Condition" : { "{condition-operator}" : { "{condition-key}" : "{condition-value}" }}
```

La clé de contexte que vous spécifiez dans une condition de politique peut être une clé de contexte de condition globale ou une clé de contexte spécifique au service. Les clés de contexte de condition globale possèdent le préfixe `aws:`. Les clés de contexte spécifiques au service possèdent le préfixe du service. Par exemple, Amazon EC2 vous permet d'écrire une condition à l'aide de la clé de contexte `ec2:InstanceType`, qui est propre à ce service. Pour connaître les clés de contexte IAM spécifiques au service ayant le préfixe `iam:`, veuillez consulter la rubrique Clés de contexte IAM et de AWS STS condition.

Les noms de clé de contexte ne sont pas sensibles à la casse. Par exemple, inclure la clé de contexte `aws:SourceIP` revient à tester `AWS:SourceIp`. La sensibilité à la casse des valeurs des clés de contexte dépend de l'opérateur de condition que vous utilisez. Par exemple, la condition suivante inclut l'opérateur `StringEquals` pour garantir que seules les demandes effectuées par `johndoe` correspondent. Les utilisateurs nommés `JohnDoe` se voient refuser l'accès.

```
"Condition" : { "StringEquals" : { "aws:username" : "johndoe" }}
```

La condition suivante utilise l'opérateur `StringEqualsIgnoreCase` pour la correspondance avec les utilisateurs nommés `johndoe` ou `JohnDoe`.

```
"Condition" : { "StringEqualsIgnoreCase" : { "aws:username" : "johndoe" }}
```

Certaines clés de contexte prennent en charge des paires clé-valeur qui vous permettent de spécifier une partie du nom de clé. Les exemples incluent la clé de `aws:RequestTag/tag-key`contexte AWS KMS `kms:EncryptionContext:encryption_context_key`, la clé de contexte et la clé de `ResourceTag/tag-key`contexte prises en charge par plusieurs services.

- Si vous utilisez la clé de contexte `ResourceTag/tag-key` pour un service comme Amazon EC2, vous devez spécifier un nom de clé pour `tag-key`.
- Les noms de clé ne sont pas sensibles à la casse. Cela signifie que si vous spécifiez `"aws:ResourceTag/TagKey1": "Value1"` dans l'élément de condition de votre politique, la condition correspond à une clé de balise de ressource nommée `TagKey1` ou `tagkey1`, mais pas aux deux.
- AWS les services qui prennent en charge ces attributs peuvent vous permettre de créer plusieurs noms de clés qui ne diffèrent qu'au cas par cas. Par exemple, vous pouvez baliser une instance Amazon EC2 avec les interfaces `ec2=test1` et `EC2=test2`. Lorsque vous utilisez une condition comme `"aws:ResourceTag/EC2": "test1"` pour autoriser l'accès à cette ressource, le nom de clé correspond aux deux balises, mais une seule valeur correspond. Cela peut entraîner des échecs de condition inattendus.

> **Important**
>
> En tant que bonne pratique, assurez-vous que les membres de votre compte suivent une convention de dénomination cohérente pour les attributs avec paire clé-valeur. Les exemples incluent les balises ou les contextes de chiffrement AWS KMS . Vous pouvez l'appliquer en utilisant la clé de `aws:TagKeys`contexte pour le balisage ou `kms:EncryptionContextKeys`pour le contexte de AWS KMS chiffrement.

- Pour obtenir une liste de tous les opérateurs de condition et une description de leur fonctionnement, veuillez consulter la rubrique Opérateurs de condition.
- Sauf indication contraire, toutes les clés de contexte peuvent comporter plusieurs valeurs. Pour savoir comment traiter des clés de contexte qui ont plusieurs valeurs, veuillez consulter la rubrique Clés de contexte à valeurs multiples.
- Pour obtenir la liste de l'ensemble des clés de contexte disponibles dans le monde entier, veuillez consulter la rubrique AWS clés contextuelles de condition globale.
- Pour les clés de contexte de condition définies par chaque service, voir Actions, ressources et clés de condition pour les AWS services.

Contexte de la demande

Lorsqu'un principal fait une demande à AWS, AWS rassemble les informations de la demande dans un contexte de demande. Ces informations servent à évaluer et autoriser la demande. Vous pouvez utiliser l'élément `Condition` d'une politique JSON pour tester des clés de contexte spécifiques par rapport au contexte de la demande. Par exemple, vous pouvez créer une politique qui utilise la clé de CurrentTime contexte aws : pour permettre à un utilisateur d'effectuer des actions dans un intervalle de dates spécifique uniquement.

Lorsqu'une demande est soumise, AWS évalue chaque clé de contexte de la politique et renvoie une valeur vraie, fausse, absente et parfois nulle (une chaîne de données vide). L'absence de clé de contexte dans la demande est considérée comme une absence de correspondance. Par exemple, la politique suivante autorise la suppression de votre propre dispositif d'authentification multifactorielle (MFA), mais uniquement si vous vous êtes connecté avec MFA au cours de la dernière heure (3 600 secondes).

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "AllowRemoveMfaOnlyIfRecentMfa",
        "Effect": "Allow",
        "Action": [
            "iam:DeactivateMFADevice"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}",
        "Condition": {
            "NumericLessThanEquals": {"aws:MultiFactorAuthAge": "3600"}
        }
    }
}
```

Le contexte de la demande peut renvoyer les valeurs suivantes :

- True : si le demandeur s'est connecté avec MFA au cours de la dernière heure ou moins, la condition renvoie la valeur true.
- False : si le demandeur s'est connecté avec MFA il y a plus d'une heure, la condition renvoie la valeur false.
- Absence : si le demandeur a fait une demande à l'aide de ses clés d'accès utilisateur IAM dans l' AWS API AWS CLI or, la clé n'est pas présente. Dans ce cas, la clé est manquante et il n'y a pas de correspondance.

- Null : pour les clés de contexte définies par l'utilisateur, telles que la transmission des balises d'une demande, il est possible d'inclure une chaîne vide. Dans ce cas, la valeur dans le contexte de la demande est null. Une valeur nulle peut renvoyer true dans certains cas. Par exemple, si vous utilisez l'opérateur de condition `ForAllValues` à valeurs multiples avec la clé de contexte `aws:TagKeys`, un résultat inattendu peut se produire si le contexte de la demande renvoie null. Pour plus d'informations, consultez aws : TagKeys etClés de contexte à valeurs multiples.

Bloc Condition

L'exemple suivant illustre le format de base d'un élément `Condition` :

```
"Condition": {"StringLike": {"s3:prefix": ["janedoe/*"]}}
```

Une valeur de la demande est représentée par une clé de contexte. Dans ce cas, il s'agit de `s3:prefix`. La valeur clé de contexte est comparée à une valeur que vous spécifiez comme valeur littérale, par exemple `janedoe/*`. Le type de comparaison à effectuer est spécifié par l'opérateur de condition (ici, `StringLike`). Vous pouvez créer des conditions qui comparent des chaînes, des dates, des numéros et autres à l'aide d'opérateurs de comparaison booléens standard comme est égal à, supérieur à ou inférieur à. Lorsque vous utilisez des opérateurs de chaîne ou des opérateurs ARN, vous pouvez également utiliser une variable de politique dans la valeur de clé de contexte. L'exemple suivant inclut la variable `aws:username`.

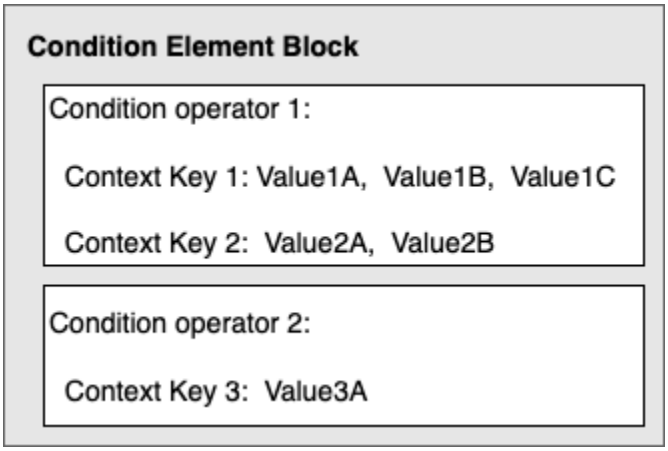
```
"Condition": {"StringLike": {"s3:prefix": ["${aws:username}/*"]}}
```

Dans certains cas, les clés de contexte peuvent contenir plusieurs valeurs. Par exemple, une demande à Amazon DynamoDB peut retourner ou mettre à jour plusieurs attributs d'une table. Une politique d'accès aux tables DynamoDB peut inclure la clé de contexte `dynamodb:Attributes`, qui contient tous les attributs spécifiés dans la demande. Vous pouvez tester les divers attributs de la demande par rapport à une liste d'attributs autorisés dans une politique à l'aide d'opérateurs de définition dans l'élément `Condition`. Pour plus d'informations, consultez Clés de contexte à valeurs multiples.

Lorsque la politique est évaluée lors d'une demande, AWS remplace la clé par la valeur correspondante de la demande. (Dans cet exemple, AWS utiliserait la date et l'heure de la demande.) L'évaluation de la condition retourne True ou False, ce qui est pris en compte pour déterminer si la politique dans sa totalité autorise ou refuse la demande.

Plusieurs valeurs dans un élément Condition

Un élément `Condition` peut contenir plusieurs opérateurs de condition, et chaque opérateur de condition peut également inclure plusieurs paires clé-valeur de contexte. L'illustration suivante décrit ce scénario.



Pour plus d'informations, voir Clés de contexte à valeurs multiples.

Éléments de politique JSON IAM : Opérateurs de condition

Utilisez les opérateurs de condition de l'élément `Condition` pour faire correspondre la clé de condition et la valeur de la politique avec les valeurs du contexte de la demande. Pour en savoir plus sur l'élément `Condition`, consultez Éléments de politique JSON IAM : Condition.

L'opérateur de condition que vous pouvez utiliser dans une politique dépend de la clé de condition que vous choisissez. Vous pouvez choisir une clé de condition globale ou une clé de condition spécifique au service. Pour savoir quel opérateur de condition vous pouvez utiliser pour une clé de condition globale, veuillez consulter AWS clés contextuelles de condition globale. Pour savoir quel opérateur de condition vous pouvez utiliser pour une clé de condition spécifique à un service, consultez Actions, ressources et clés de condition pour les AWS services et choisissez le service que vous souhaitez consulter.

> **Important**
>
> Si la clé que vous spécifiez dans une condition de stratégie n'est pas présente dans le contexte de la requête, les valeurs ne correspondent pas et la condition est fausse. Si la condition de stratégie requiert qu'il n'y ait aucune correspondance de clé, tels que `StringNotLike` ou `ArnNotLike` et la touche de droite n'est pas présente, la condition est

vraie. Cette logique s'applique à tous les opérateurs de condition sauf... IfExistset Contrôle nul. Ces opérateurs testent si la clé est présente (existe) dans le contexte de demande.

Les opérateurs de condition peuvent être regroupés dans les catégories suivantes :

- String
- Numérique
- Date et heure
- Booléen
- Binaire
- Adresse IP
- Amazon Resource Name (ARN) (disponible uniquement pour certains services.)
- ... IfExists(vérifie si la valeur clé existe dans le cadre d'une autre vérification)
- Null check (vérifie si la valeur de clé existe en tant que vérification autonome)

Opérateurs de condition de chaîne

Les opérateurs de condition de chaîne permettent de créer des éléments `Condition` qui limitent l'accès après comparaison d'une clé à une valeur de chaîne.

Opérateur de condition	Description
`StringEquals`	Correspondance exacte, respect de la casse
`StringNotEquals`	Correspondance négative
`StringEqualsIgnoreCase`	Correspondance exacte, non respect de la casse
`StringNotEqualsIgnoreCase`	Correspondance négative, non respect de la casse
`StringLike`	Correspondance avec respect de la casse. Les valeurs peuvent inclure un caractère générique (*) correspondant à plusieurs caractères et un caractère générique (?) correspondant à

Opérateur de condition	Description
	un seul caractère n'importe où dans la chaîne. Vous devez spécifier des caractères génériques pour obtenir des correspon dances de chaînes partielles. **Note** Si une clé contient plusieurs valeurs, `StringLik e` peut être qualifié avec les opérateurs d'ensembl e `ForAllValues:StringLike` et `ForAnyVal ue:StringLike` . Pour plus d'informations, consultez Clés de contexte à valeurs multiples.
`StringNotLike`	Correspondance avec non respect de la casse. Les valeurs peuvent inclure un caractère générique (*) correspondant à plusieurs caractères ou un caractère générique (?) correspon dant à un seul caractère n'importe où dans la chaîne.

Par exemple, l'instruction suivante contient un élément `Condition` qui utilise la clé `aws:PrincipalTag` pour spécifier que le principal qui fait la requête doit être balisé avec la catégorie de tâche `iamuser-admin` .

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/*",
    "Condition": {"StringEquals": {"aws:PrincipalTag/job-category": "iamuser-admin"}}
  }
}
```

Si la clé que vous spécifiez dans une condition de politique n'est pas présente dans le contexte de demande, les valeurs ne correspondent pas. Dans cet exemple, la clé `aws:PrincipalTag/job-category` est présente dans le contexte de demande si le principal utilise un utilisateur IAM avec des balises attachées. Elle est également incluse pour un principal utilisant un rôle IAM avec des balises ou des balises de session attachées. Si un utilisateur sans la balise tente d'afficher ou de

modifier une clé d'accès, la condition renvoie `false` et la demande est implicitement refusée par cette instruction.

Vous pouvez utiliser une variable de politique avec l'opérateur de condition `String`.

L'exemple suivant utilise l'opérateur de condition `StringLike` pour établir une correspondance entre une chaîne et une variable de politique afin de créer une politique qui permet à un utilisateur IAM de se servir de la console Amazon S3 pour gérer son propre « répertoire de base » dans un compartiment Amazon S3. La politique autorise les actions spécifiées dans un compartiment S3 si l'élément `s3:prefix` correspond à l'un des modèles spécifiés.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::BUCKET-NAME",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}",
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}/*"
      ]
    }
  ]
}
```

Pour un exemple de politique qui montre comment utiliser l'`Condition`élément pour restreindre l'accès aux ressources en fonction d'un ID d'application et d'un ID utilisateur pour la fédération OIDC, voirAmazon S3 : permet aux utilisateurs Amazon Cognito d'accéder aux objets dans leur compartiment.

Correspondance des caractères génériques

Les opérateurs de condition de chaîne réalisent une correspondance sans modèle qui n'applique pas de format prédéfini. Les opérateurs de condition ARN et de condition de date constituent un sous-ensemble d'opérateurs de chaîne qui appliquent une structure à la valeur de la clé de condition. Lorsque vous utilisez des StringNotLike opérateurs StringLike or pour des correspondances partielles de chaîne d'un ARN ou d'une date, la correspondance ignore quelle partie de la structure est associée à un caractère générique.

Par exemple, les conditions suivantes recherchent une correspondance partielle d'un ARN à l'aide de différents opérateurs de condition.

Lorsqu'il ArnLike est utilisé, les parties partition, service, identifiant de compte, type de ressource et identifiant de ressource partiel de l'ARN doivent correspondre exactement à l'ARN dans le contexte de la demande. Seuls la région et le chemin de ressource permettent une correspondance partielle.

```
"Condition": {"ArnLike": {"aws:SourceArn": "arn:aws:cloudtrail:*:111122223333:trail/
*"}}
```

Lorsqu'elle StringLike est utilisée à la place de ArnLike, la correspondance ignore la structure de l'ARN et permet une correspondance partielle, quelle que soit la partie associée au caractère générique.

```
"Condition": {"StringLike": {"aws:SourceArn": "arn:aws:cloudtrail:*:111122223333:trail/
*"}}
```

ARN	ArnLike	StringLike
arn:aws:cloudtrail:us-west-2:111122223333:trail/finance	Correspondance	Correspondance
arn:aws:cloudtrail:us-east-2:111122223333:trail/finance/archive	Correspondance	Correspondance

ARN	ArnLike	StringLike
arn:aws:cloudtrail:us-east-2:444455556666:user/111122223333:trail/finance	Aucune correspondance	Correspondance

Opérateurs de condition numériques

Les opérateurs de condition numériques permettent de créer des éléments `Condition` qui limitent l'accès après comparaison d'une clé à un entier ou une valeur décimale.

Opérateur de condition	Description
`NumericEquals`	Correspondance
`NumericNotEquals`	Correspondance négative
`NumericLessThan`	Correspondance « Inférieur à »
`NumericLessThanEquals`	Correspondance « Inférieur ou égal à »
`NumericGreaterThan`	Correspondance « Supérieur à »
`NumericGreaterThanEquals`	Correspondance « Supérieur ou égal à »

Par exemple, l'instruction suivante contient un élément `Condition` qui utilise l'opérateur de condition `NumericLessThanEquals` avec la clé `s3:max-keys` pour spécifier que le demandeur peut répertorier jusqu'à 10 objets dans `example_bucket` à la fois.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket",
```

```
    "Condition": {"NumericLessThanEquals": {"s3:max-keys": "10"}}
  }
}
```

Si la clé que vous spécifiez dans une condition de politique n'est pas présente dans le contexte de demande, les valeurs ne correspondent pas. Dans cet exemple, la clé `s3:max-keys` est toujours présente dans la demande lorsque vous effectuez l'opération `ListBucket`. Si cette politique autorisait toutes les opérations Amazon S3, seules les opérations incluant la clé de contexte `max-keys` avec une valeur inférieure ou égale à 10 seraient autorisées.

Vous ne pouvez pas utiliser une variable de politique avec l'opérateur de condition `Numeric`.

Opérateurs de condition de date

Les opérateurs de condition de date permettent de créer des éléments `Condition` qui limitent l'accès après comparaison d'une clé à une valeur date/heure. Vous pouvez utiliser ces opérateurs de condition avec la clé `aws:CurrentTime` ou `aws:EpochTime`. Vous devez spécifier les valeurs date/heure à l'aide de l'une des implémentations W3C des formats de date ISO 8601 ou du format d'époque (UNIX).

> **Note**
>
> Les caractères génériques ne sont pas autorisés dans les opérateurs de condition de date.

Opérateur de condition	Description
`DateEquals`	Correspondance à une date spécifique
`DateNotEquals`	Correspondance négative
`DateLessThan`	Correspondance avant une date et heure spécifiques
`DateLessThanEquals`	Correspondance à ou avant une date et heure spécifiques
`DateGreaterThan`	Correspondance après une date et heure spécifiques
`DateGreaterThanEqu als`	Correspondance à ou après une date et heure spécifiques

Par exemple, l'instruction suivante contient un élément `Condition` qui utilise l'opérateur de condition `DateGreaterThan` avec la clé `aws:TokenIssueTime`. Cette condition spécifie que les informations d'identification de sécurité temporaires utilisées pour effectuer la demande ont été publiées en 2020. Cette politique peut être mise à jour par programme tous les jours pour s'assurer que les membres du compte utilisent de nouvelles informations d'identification.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/*",
    "Condition": {"DateGreaterThan": {"aws:TokenIssueTime": "2020-01-01T00:00:01Z"}}
  }
}
```

Si la clé que vous spécifiez dans une condition de politique n'est pas présente dans le contexte de demande, les valeurs ne correspondent pas. La clé `aws:TokenIssueTime` n'est présente dans le contexte de demande que lorsque le principal utilise des informations d'identification temporaires pour effectuer la demande. La clé n'est pas présente dans AWS CLI les demandes AWS d'API ou de AWS SDK effectuées à l'aide de clés d'accès. Dans cet exemple, si un utilisateur IAM tente d'afficher ou de modifier une clé d'accès, la demande est refusée.

Vous ne pouvez pas utiliser une variable de politique avec l'opérateur de condition `Date`.

Opérateurs de condition booléens

Les opérateurs de condition booléens permettent de créer des éléments `Condition` qui limitent l'accès après comparaison d'une clé à « true » ou « false ».

Opérateur de condition	Description
`Bool`	Correspondance booléenne

Par exemple, cette politique basée sur l'identité utilise l'opérateur de condition `Bool` avec la clé `aws:SecureTransport` pour refuser la réplication d'objets et de balises d'objets vers le compartiment de destination et son contenu si la demande n'est pas effectuée via SSL.

> **Important**
>
> Cette politique ne permet aucune action. Utilisez cette stratégie conjointement à d'autres stratégies qui autorisent des actions spécifiques.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "BooleanExample",
      "Action": "s3:ReplicateObject",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ],
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "false"
        }
      }
    }
  ]
}
```

Si la clé que vous spécifiez dans une condition de politique n'est pas présente dans le contexte de demande, les valeurs ne correspondent pas. Le contexte de la demande `aws:SecureTransport` renvoie true ou false.

Vous pouvez utiliser une variable de politique avec l'opérateur de condition `Boolean`.

Opérateurs de condition binaires

L'opérateur de condition `BinaryEquals` permet de créer des éléments `Condition` qui analysent les valeurs de clé qui utilisent un format binaire. Il compare la valeur de la clé spécifiée, octet par octet, à une représentation encodée au format Base64 de la valeur binaire dans la politique.

```
"Condition" : {
  "BinaryEquals": {
    "key" : "QmluYXJ5VmFsdWVJbkJhc2U2NA=="
```

```
  }
}
```

Si la clé que vous spécifiez dans une condition de politique n'est pas présente dans le contexte de demande, les valeurs ne correspondent pas.

Vous ne pouvez pas utiliser une variable de politique avec l'opérateur de condition `Binary`.

Opérateurs de condition d'adresse IP

Les opérateurs de condition d'adresse IP permettent de créer des éléments `Condition` qui limitent l'accès après comparaison d'une clé à une adresse IPv4 ou IPv6 ou une plage d'adresses IP. Vous utilisez ces opérateurs avec la clé `aws:SourceIp`. La valeur doit utiliser au format CIDR standard (par exemple, 203.0.113.0/24 ou 2001:DB8:1234:5678::/64). Si vous spécifiez une adresse IP sans préfixe de routage associé, IAM utilise la valeur de préfixe par défaut `/32`.

Certains AWS services prennent en charge le protocole IPv6, en utilisant : : pour représenter une plage de 0. Pour savoir si un service prend en charge IPv6, consultez la documentation correspondante.

Opérateur de condition	Description
`IpAddress`	Adresse IP ou plage d'adresses IP spécifiée
`NotIpAddress`	Toutes les adresses IP à l'exception de l'adresse IP ou de la plage d'adresse IP spécifiée

Par exemple, l'instruction suivante utilise l'opérateur de condition `IpAddress` avec la clé `aws:SourceIp` pour spécifier que la demande doit provenir d'une adresse IP comprise dans la plage 203.0.113.0 à 203.0.113.255.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/*",
    "Condition": {"IpAddress": {"aws:SourceIp": "203.0.113.0/24"}}
  }
```

```
}
```

La clé de condition `aws:SourceIp` est résolue à l'aide de l'adresse IP d'où provient la demande. Si la demande provient d'une instance Amazon EC2, `aws:SourceIp` correspond à l'adresse IP publique de l'instance.

Si la clé que vous spécifiez dans une condition de politique n'est pas présente dans le contexte de demande, les valeurs ne correspondent pas. La clé `aws:SourceIp` figure toujours dans le contexte de demande, sauf lorsque le demandeur utilise un point de terminaison VPC pour effectuer la demande. Dans ce cas, la condition renvoie `false` et la demande est implicitement refusée par cette instruction.

Vous ne pouvez pas utiliser une variable de politique avec l'opérateur de condition `IpAddress`.

L'exemple suivant montre comment combiner les adresses IPv4 et IPv6 pour couvrir la totalité des adresses IP valides de l'organisation. Nous recommandons de mettre à jour les politiques de l'organisation par vos plages d'adresses IPv6 (en plus des plages d'adresses IPv4 dont vous disposez déjà) pour être sûr que les politiques continuent à fonctionner lors de la transition vers IPv6.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "someservice:*",
    "Resource": "*",
    "Condition": {
      "IpAddress": {
        "aws:SourceIp": [
          "203.0.113.0/24",
          "2001:DB8:1234:5678::/64"
        ]
      }
    }
  }
}
```

La clé de condition `aws:SourceIp` fonctionne uniquement dans une politique JSON si vous appelez l'API que vous testez directement en tant qu'utilisateur. En revanche, si vous utilisez un service pour appeler le service cible en votre nom, ce service voit l'adresse IP du service appelant plutôt que celle de l'utilisateur d'origine. Cela peut se produire, par exemple, si vous avez l'habitude AWS

CloudFormation d'appeler Amazon EC2 pour créer des instances pour vous. Actuellement, il n'est pas possible de transmettre l'adresse IP d'origine au service cible via un service appelant à des fins d'évaluation dans une politique JSON. Pour ces types d'appels d'API de service, vous ne devez pas utiliser la clé de condition `aws:SourceIp`.

Opérateurs de condition d'Amazon Resource Name (ARN)

Les opérateurs de condition d'Amazon Resource Name (ARN) permettent de créer des éléments `Condition` qui limitent l'accès après comparaison d'une clé à un ARN. L'ARN est considéré comme étant une chaîne.

Opérateur de condition	Description
`ArnEquals` , `ArnLike`	Correspondance à l'ARN avec respect de la casse. Chacun des six composants de l'ARN, séparés par deux points, est vérifié séparément et chacun peut inclure un caractère générique correspondant à plusieurs caractères (*) ou un caractère générique correspondant à un caractère (?). Les opérateurs de condition `ArnEquals` et `ArnLike` se comportent de manière identique.
`ArnNotEquals` , `ArnNotLike`	Correspondance négative à l'ARN. Les opérateurs de condition `ArnNotEquals` et `ArnNotLike` se comportent de manière identique.

Vous pouvez utiliser une variable de politique avec l'opérateur de condition `ARN`.

L'exemple de politique basée sur les ressources suivant montre une politique attachée à une file d'attente Amazon SQS à laquelle vous souhaitez envoyer des messages SNS. Elle donne l'autorisation à Amazon SNS d'envoyer des messages à une ou plusieurs files d'attente de votre choix, mais uniquement si le service envoie le message pour le compte d'une ou plusieurs rubriques Amazon SNS spécifiques. Vous spécifiez la file d'attente dans le champ `Resource`, tandis que la rubrique Amazon SNS est la valeur de la clé `SourceArn`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "123456789012"},
```

```
    "Action": "SQS:SendMessage",
    "Resource": "arn:aws:sqs:REGION:123456789012:QUEUE-ID",
    "Condition": {"ArnEquals": {"aws:SourceArn":
 "arn:aws:sns:REGION:123456789012:TOPIC-ID"}}
  }
}
```

Si la clé que vous spécifiez dans une condition de politique n'est pas présente dans le contexte de demande, les valeurs ne correspondent pas. La clé `aws:SourceArn` figure dans le contexte de demande uniquement si une ressource déclenche un service pour appeler un autre service au nom du propriétaire de la ressource. Si un utilisateur IAM tente d'effectuer cette opération directement, la condition renvoie `false` et la demande est implicitement refusée par cette instruction.

... IfExists opérateurs de conditions

Vous pouvez ajouter `IfExists` à la fin de n'importe quel nom d'opérateur de condition, à l'exception de la condition `Null`, par exemple,`StringLikeIfExists`. Ceci équivaut à spécifier que « Si la clé de politique est présente dans le contexte de la demande, la clé doit être traitée comme spécifié dans la politique. Si la clé n'est pas présente, la condition évalue l'élément de condition comme vrai. » Les autres éléments de condition dans l'instruction peuvent toujours se traduire par une absence de correspondance, mais pas par une clé manquante lors de la vérification avec `...IfExists`. Si vous utilisez un élément `"Effect": "Deny"` avec un opérateur de condition négatif tel que `StringNotEqualsIfExists`, la demande est toujours refusée, même s'il manque la balise.

Exemple d'utilisation de **`IfExists`**

De nombreuses clés de condition contiennent des informations se rapportant à un type spécifique de ressources et elles ne sont présentes que lorsque vous accédez à ce type de ressources. Ces clés de condition n'existent pas pour les autres types de ressources. Le fait que l'instruction ne s'applique qu'à un type spécifique de ressources ne pose pas problème. Toutefois, dans certains scénarios, une même instruction peut s'appliquer à plusieurs types de ressources, par exemple lorsque l'instruction de politique référence les actions de plusieurs services ou lorsqu'une action donnée d'un service accède à différents types de ressources dans un même service. Dans ce cas, l'inclusion d'une clé de condition applicable uniquement à une des ressources dans l'instruction de politique peut provoquer l'échec de l'élément `Condition` et de ce fait, l'élément `"Effect"` ne s'applique pas.

Prenons l'exemple de politique suivant :

```
{
```

```
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "THISPOLICYDOESNOTWORK",
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": "*",
    "Condition": {"StringLike": {"ec2:InstanceType": [
      "t1.*",
      "t2.*",
      "m3.*"
    ]}}
  }
}
```

L'intention de la politique précédente est de permettre à l'utilisateur de lancer n'importe quelle instance de type `t1`, `t2` ou `m3`. Toutefois, le lancement d'une instance requiert non seulement l'accès à l'instance proprement dite, mais également à de nombreuses ressources telles que des images, des paires de clés, des groupes de sécurité, etc. L'ensemble de l'instruction est évalué par rapport à chaque ressource requise pour le lancement de l'instance. Ces ressources supplémentaires n'ont pas la clé de condition `ec2:InstanceType` et, par conséquent, la vérification `StringLike` échoue et l'utilisateur n'est autorisé à lancer aucun type d'instance.

Pour éviter ce problème, utilisez l'opérateur de condition `StringLikeIfExists` à la place. De cette façon, le test n'est effectué que si la clé de condition existe. Cela peut être interprété comme suit : « Si la ressource en cours de vérification est dotée d'une clé de condition `ec2:InstanceType`, l'action peut uniquement être autorisée si la valeur de la clé commence par `t1.`, `t2.` ou `m3.`. Si la ressource en cours de vérification n'est pas dotée de cette clé de condition, peu importe. » L'astérisque (*) figurant dans les valeurs des clés de condition, lorsqu'il est utilisé avec l'opérateur de condition `StringLikeIfExists`, est interprété comme un caractère générique pour obtenir des correspondances de chaînes partielles. L'instruction `DescribeActions` inclut les actions requises pour afficher l'instance dans la console.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RunInstance",
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "*",
            "Condition": {
```

```
                "StringLikeIfExists": {
                    "ec2:InstanceType": [
                        "t1.*",
                        "t2.*",
                        "m3.*"
             ]}}
        },
        {
            "Sid": "DescribeActions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "ec2:DescribeInstances",
                "ec2:DescribeVpcs",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups"
            ],
            "Resource": "*"
        }]
}
```

Opérateur de condition pour vérifier l'existence de clés de condition

Utilisez un opérateur de condition `Null` pour vérifier si une clé de condition est absente au moment de l'autorisation. Dans l'instruction de politique, utilisez `true` (la clé n'existe pas ; elle est nulle) ou `false` (la clé existe et sa valeur n'est pas nulle).

Vous ne pouvez pas utiliser une variable de politique avec l'opérateur de condition `Null`.

Par exemple, vous pouvez utiliser cet opérateur de condition pour déterminer si un utilisateur effectue l'opération à l'aide de ses propres informations d'identification ou d'informations d'identification temporaires. S'il utilise des informations d'identification temporaires, la clé `aws:TokenIssueTime` existe et elle est dotée d'une valeur. L'exemple suivant illustre une condition qui spécifie que l'utilisateur ne doit pas avoir recours à des informations d'identification temporaires (la clé ne doit pas exister) pour exécuter l'API Amazon EC2.

```
{
  "Version": "2012-10-17",
  "Statement":{
      "Action":"ec2:*",
      "Effect":"Allow",
```

```
      "Resource":"*",
      "Condition":{"Null":{"aws:TokenIssueTime":"true"}}
  }
}
```

Conditions avec plusieurs clés ou valeurs de contexte

Vous pouvez utiliser l'élément `Condition` d'une politique pour tester plusieurs clés ou plusieurs valeurs de contexte pour une seule clé de contexte dans une demande. Lorsque vous faites une demande AWS, par programmation ou par le biais du AWS Management Console, votre demande inclut des informations sur votre principal, votre fonctionnement, vos balises, etc. Vous pouvez utiliser des clés de contexte pour tester les valeurs des clés de contexte correspondantes dans la demande, avec les clés de contexte spécifiées dans la condition de politique. Pour en savoir plus sur les informations et les données incluses dans une demande, consultez Contexte de la demande.

Rubriques

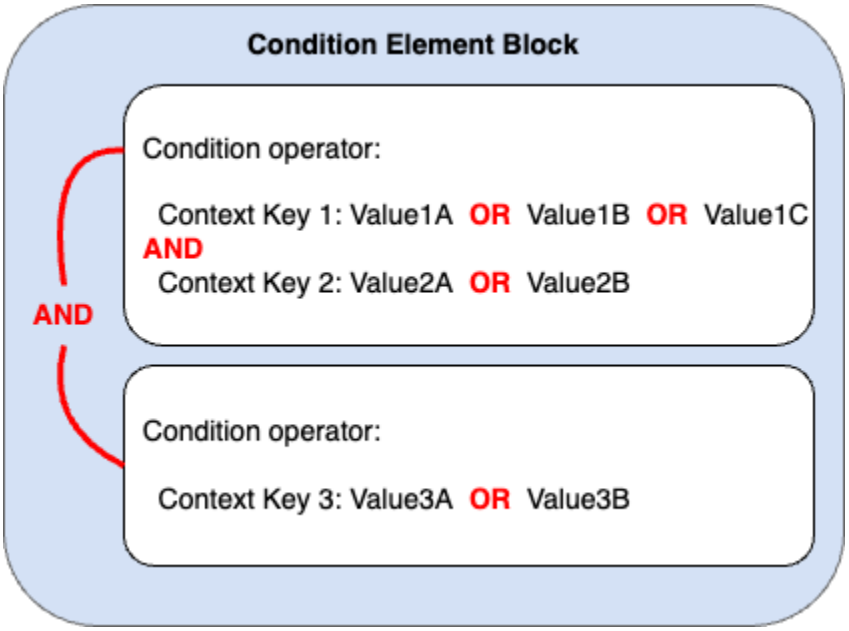
- Logique d'évaluation pour plusieurs clés ou valeurs de contexte
- Logique d'évaluation des opérateurs de condition de correspondance négative

Logique d'évaluation pour plusieurs clés ou valeurs de contexte

Un élément `Condition` peut contenir plusieurs opérateurs de condition, et chaque opérateur de condition peut également inclure plusieurs paires clé-valeur de contexte. La plupart des clés de contexte prennent en charge l'utilisation de plusieurs valeurs, sauf indication contraire.

- Si votre instruction de politique dispose de plusieurs opérateurs de condition, ils sont évalués à l'aide d'un opérateur logique `AND`.
- Si votre instruction de politique dispose de plusieurs clés de contexte attachées à un même opérateur de condition, celles-ci sont évaluées à l'aide d'un opérateur logique `AND`.
- Si un même opérateur de condition comprend plusieurs valeurs pour une clé de contexte, ces valeurs sont évaluées à l'aide d'un opérateur logique `OR`.
- Si un même opérateur de condition de correspondance négative comprend plusieurs valeurs pour une clé de contexte, ces valeurs sont évaluées à l'aide d'un opérateur logique `NOR`.

Toutes les clés de contexte d'un bloc d'éléments de condition doivent être résolues sur true (vrai) pour invoquer l'effet `Allow` ou `Deny` souhaité. La figure suivante illustre la logique d'évaluation d'une condition comportant plusieurs opérateurs de condition et des paires clé-valeur de contexte.



Par exemple, la politique de compartiment S3 suivante illustre la manière dont la figure précédente est représentée dans une politique. Le bloc de conditions comprend les opérateurs de condition `StringEquals` et `ArnLike`, et les clés de contexte `aws:PrincipalTag` et `aws:PrincipalArn`. Afin d'invoquer l'effet `Allow` ou `Deny` souhaité, toutes les clés de contexte du bloc de conditions doivent être résolues sur true (vrai). L'utilisateur qui fait la demande doit disposer des deux clés de balise du principal, department et role, qui incluent l'une des valeurs de clés de balise spécifiées dans la politique. De plus, l'ARN principal de l'utilisateur qui fait la demande doit correspondre à l'une des valeurs `aws:PrincipalArn` spécifiées dans la politique pour être évalué comme true (vrai).

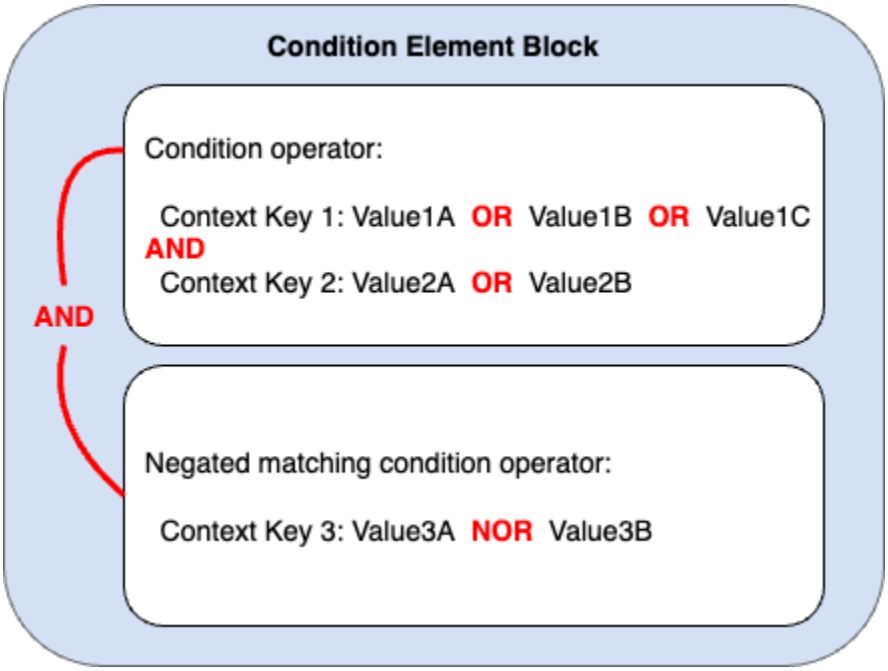
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
            "finance",
            "hr",
```

```
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
}
```

Logique d'évaluation des opérateurs de condition de correspondance négative

Certains opérateurs de condition, tels que `StringNotEquals` ou `ArnNotLike`, utilisent la correspondance négative pour comparer les paires clé-valeur de contexte de votre politique aux paires clé-valeur de contexte d'une demande. Lorsque plusieurs valeurs sont spécifiées pour une même clé de contexte dans une politique avec des opérateurs de condition de correspondance négative, les autorisations effectives fonctionnent comme un opérateur logique `NOR`. Dans le cas d'une correspondance négative, un opérateur logique `NOR` ou `NOT OR` renvoie la valeur true (vrai) uniquement si toutes les valeurs sont évaluées comme false (fausses).

La figure suivante illustre la logique d'évaluation d'une condition comportant plusieurs opérateurs de condition et des paires clé-valeur de contexte. La figure inclut un opérateur de condition de correspondance négative pour la clé de contexte 3.



Par exemple, la politique de compartiment S3 suivante illustre la manière dont la figure précédente est représentée dans une politique. Le bloc de conditions comprend les opérateurs de condition `StringEquals` et `ArnNotLike`, et les clés de contexte `aws:PrincipalTag` et `aws:PrincipalArn`. Afin d'invoquer l'effet `Allow` ou `Deny` souhaité, toutes les clés de contexte du bloc de conditions doivent être résolues sur true (vrai). L'utilisateur qui fait la demande doit disposer des deux clés de balise du principal, department et role, qui incluent l'une des valeurs de clés de balise spécifiées dans la politique. Puisque l'opérateur de condition `ArnNotLike` utilise la correspondance négative, l'ARN principal de l'utilisateur qui fait la demande ne doit pas correspondre à l'une des valeurs `aws:PrincipalArn` spécifiées dans la politique pour être évalué comme true (vrai).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
      "Condition": {
        "StringEquals": {
```

```
          "aws:PrincipalTag/department": [
            "finance",
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnNotLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
}
```

Clés de contexte à valeur unique ou à valeurs multiples

La différence entre les clés de contexte à valeur unique et à valeurs multiples dépend du nombre de valeurs dans le contexte de la demande, et non du nombre de valeurs dans la condition de la politique.

- Les clés de contexte de condition à valeur unique ont au plus une valeur dans le contexte de la demande. Par exemple, vous pouvez baliser des ressources dans AWS. Les balises de ressources sont stockées sous forme de paires clé-valeur de balise. Une clé de balise de ressource peut avoir une seule valeur de balise. Par conséquent, the section called “ResourceTag” est une clé de contexte à valeur unique. N'utilisez pas d'opérateur d'ensemble de conditions avec une clé de contexte à valeur unique.
- Les clés de contexte de condition à valeurs multiples peuvent comporter plusieurs valeurs dans le contexte de la demande. Par exemple, vous pouvez étiqueter des ressources AWS et inclure plusieurs paires clé-valeur de balise dans une demande. Ainsi, the section called “TagKeys” est une clé de contexte à valeurs multiples. Les clés de contexte à valeurs multiples nécessitent un opérateur d'ensemble de conditions.

> **Important**
>
> Les clés de contexte à valeurs multiples nécessitent un opérateur d'ensemble de conditions. N'utilisez pas d'opérateurs d'ensemble de conditions `ForAllValues` ou `ForAnyValue` avec des clés de contexte à valeur unique. Pour en savoir plus sur les opérateurs d'ensemble de conditions, veuillez consulter la rubrique Clés de contexte à valeurs multiples.

Les classifications À valeur unique et À valeurs multiples sont incluses dans la description de chaque clé de contexte de condition, dans le Type de valeur de la rubrique AWS clés contextuelles de condition globale. La Référence de l'autorisation de service utilise une classification de type de valeur différente pour les clés de contexte à valeurs multiples au format suivant : un préfixe `ArrayOf` suivi du type de catégorie de l'opérateur de condition. Par exemple, `ArrayOfString` ou `ArrayOfARN`.

Par exemple, une demande peut provenir d'un seul point de terminaison d'un VPC au maximum. the section called “SourceVpce” est donc une clé de contexte à valeur unique. Étant donné qu'un service peut avoir plus d'un nom de principal de service qui appartient au service, lois : PrincipalService NamesList est une clé de contexte à valeurs multiples.

Vous pouvez utiliser n'importe quelle clé de contexte à valeur unique disponible en tant que variable de politique. Vous ne pouvez pas utiliser de clé de contexte à valeurs multiples en tant que variable de politique. Pour de plus amples informations sur les variables de politique, veuillez consulter Éléments des politiques IAM : variables et balises.

Les clés de contexte à valeurs multiples nécessitent les opérateurs d'ensemble de conditions `ForAllValues` ou `ForAnyValue`. Les clés de contexte qui comprennent des paires clé-valeur telles que the section called “RequestTag” et the section called “ResourceTag” peuvent prêter à confusion, car il peut y avoir plusieurs valeurs *`tag-key`*. Mais comme chaque *`tag-key`* ne peut avoir qu'une seule valeur, `aws:RequestTag` et `aws:ResourceTag` sont tous deux des clés de contexte à valeur unique. L'utilisation d'opérateurs d'ensembles de conditions avec des clés de contexte à valeur unique peut entraîner des politiques trop permissives.

Clés de contexte à valeurs multiples

Pour comparer votre clé de contexte de condition à un contexte de demande à valeurs multiples, vous devez utiliser les opérateurs d'ensemble `ForAllValues` ou `ForAnyValue`. Ces opérateurs d'ensembles sont utilisés pour comparer deux ensembles de valeurs, tels que l'ensemble de balises dans une demande et l'ensemble de balises dans une condition de politique.

Ces qualificatifs `ForAllValues` et `ForAnyValue` ajoutent une fonctionnalité d'opération d'ensemble à l'opérateur de condition afin que vous puissiez tester les clés de contexte avec plusieurs valeurs par rapport à plusieurs valeurs de clé de contexte dans une condition de politique. En outre, si vous incluez une clé de contexte à valeurs multiples dans votre politique avec un caractère générique ou une variable, vous devez également utiliser l'opérateur de condition `StringLike`. Les valeurs de clé de condition multiples doivent être placées entre crochets, comme dans un tableau. Par exemple, `"Key2":["Value2A", "Value2B"]`.

- `ForAllValues` : ce qualificateur teste si la valeur de chaque membre de la demande est un sous-ensemble de l'ensemble de clé de contexte de condition. La condition renvoie la valeur true (vrai) si chaque valeur de clé de contexte de la demande correspond à au moins une valeur de clé de contexte de la politique. Elle renvoie également la valeur true si la demande ne comprend pas de clés de contexte ou si la valeur de clé de contexte aboutit à un jeu de données nul, tel qu'une chaîne vide. Pour éviter que des clés de contexte manquantes ou des clés de contexte contenant des valeurs vides ne soient considérées comme true (vraies), vous pouvez inclure l'opérateur de condition `Null` dans votre politique avec une valeur fausse pour vérifier si la clé de contexte existe et si sa valeur n'est pas nulle.

  > **Important**
  >
  > Faites preuve de vigilance si vous utilisez `ForAllValues` avec un effet `Allow`, car cela peut être trop permissif si la présence de clés de contexte manquantes ou de clés de contexte avec des valeurs vides dans le contexte de la demande est inattendue. Vous pouvez inclure l'opérateur de condition `Null` dans votre politique avec une valeur fausse pour vérifier si la clé de contexte existe et si sa valeur n'est pas nulle. Pour obtenir un exemple, consultez Contrôle de l'accès en fonction des clés de balise.

- `ForAnyValue` : ce qualificateur teste si au moins un membre de l'ensemble des valeurs de clé de contexte de la demande correspond à au moins un membre de l'ensemble des valeurs de clé de contexte de votre condition de politique. La clé de contexte renvoie la valeur true si l'une des valeurs de clé de contexte de la demande correspond à l'une des valeurs de clé de contexte de la politique. Si aucune clé de contexte ne correspond ou si le jeu de données est inexistant (null), la condition renvoie la valeur false.

> **Note**
>
> La différence entre les clés de contexte à valeur unique et à valeurs multiples dépend du nombre de valeurs dans le contexte de la demande, et non du nombre de valeurs dans la condition de la politique.

Exemples de politiques de conditions

Dans les politiques IAM, vous pouvez spécifier plusieurs valeurs pour les clés de contexte à valeur unique et à valeurs multiples à des fins de comparaison avec le contexte de la demande. L'ensemble suivant d'exemples de politiques montre les conditions de politique avec plusieurs clés et valeurs de contexte.

> **Note**
>
> Si vous souhaitez envoyer une politique à inclure dans ce guide de référence, utilisez le bouton Commentaire situé au bas de cette page. Pour accéder à des exemples de politiques IAM basées sur l'identité, veuillez consulter la rubrique Exemples de politiques basées sur l'identité IAM.

Exemples de politiques de conditions : clés de contexte à valeur unique

- Plusieurs blocs de conditions avec des clés de contexte à valeur unique. (Voir cet exemple.)
- Un bloc de conditions avec plusieurs clés et valeurs de contexte à valeur unique. (Voir cet exemple.)

Exemples de politiques de conditions : clés de contexte à valeurs multiples.

- Politique de refus avec opérateur d'ensemble de conditions `ForAllValues`. (Voir cet exemple.)
- Politique de refus avec opérateur d'ensemble de conditions `ForAnyValue`. (Voir cet exemple.)

Exemples de clés de contexte à valeurs multiples

L'ensemble suivant d'exemples de politiques montre comment créer des conditions de politique à l'aide de clés de contexte à valeurs multiples.

Exemple : politique de refus avec opérateur de jeu de conditions ForAllValues

L'exemple suivant de politique basée sur l'identité interdit l'utilisation d'actions de balisage IAM lorsque des préfixes de clé de balise spécifiques sont inclus dans la demande. Chaque valeur pour la clé de contexte `aws:TagKeys` inclut un caractère générique (*) pour la correspondance partielle des chaînes. La politique inclut l'opérateur d'ensemble `ForAllValues` avec la clé de contexte `aws:TagKeys`, car la clé de contexte de la demande peut inclure plusieurs valeurs. Afin que la clé de contexte `aws:TagKeys` renvoie la valeur true, chaque valeur de la demande doit correspondre à au moins une valeur de la politique.

L'opérateur d'ensemble `ForAllValues` renvoie également la valeur true si la demande ne comprend pas de clés de contexte ou si la valeur de clé de contexte aboutit à un jeu de données nul, tel qu'une chaîne vide. Pour éviter que des clés de contexte manquantes ou des clés de contexte contenant des valeurs vides ne soient considérées comme true, incluez l'opérateur de condition `Null` dans votre politique avec une valeur `false` pour vérifier si la clé de contexte de la demande existe et si sa valeur n'est pas nulle.

> **Important**
>
> Cette politique ne permet aucune action. Utilisez cette stratégie conjointement à d'autres stratégies qui autorisent des actions spécifiques.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRestrictedTags",
      "Effect": "Deny",
      "Action": [
        "iam:Tag*",
        "iam:UnTag*"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "Null": {
          "aws:TagKeys": "false"
        },
```

```
        "ForAllValues:StringLike": {
          "aws:TagKeys": [
            "key1*",
            "key2*",
            "key3*"
          ]
        }
      }
    }
  ]
}
```

Exemple : politique de refus avec opérateur de jeu de conditions ForAnyValue

L'exemple de politique basée sur l'identité suivant interdit la création d'instantanés de volumes d'instance EC2 si des instantanés sont balisés à l'aide de l'une des clés de balise spécifiées dans la politique, `environment` ou `webserver`. La politique inclut l'opérateur d'ensemble `ForAnyValue` avec la clé de contexte `aws:TagKeys`, car la clé de contexte de la demande peut inclure plusieurs valeurs. Si votre demande de balisage inclut l'une des valeurs de clés de balise spécifiées dans la politique, la clé de contexte `aws:TagKeys` renvoie la valeur true en invoquant l'effet de la politique de refus.

> **Important**
>
> Cette politique ne permet aucune action. Utilisez cette stratégie conjointement à d'autres stratégies qui autorisent des actions spécifiques.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ec2:CreateSnapshot",
                "ec2:CreateSnapshots"
            ],
            "Resource": "arn:aws:ec2:us-west-2::snapshot/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": ["environment", "webserver"]
```

```
                }
            }
        }
    ]
}
```

Exemples de stratégies de clé de contexte à valeur unique

L'ensemble suivant d'exemples de politiques montre comment créer des conditions de politique à l'aide de clés de contexte à valeur unique.

Exemple : plusieurs blocs de conditions avec des clés de contexte à valeur unique

Lorsqu'un bloc de conditions comporte plusieurs conditions, chacune associée à une seule clé de contexte, toutes les clés de contexte doivent aboutir à la valeur true pour que l'effet `Allow` ou `Deny` souhaité soit invoqué. Lorsque vous utilisez des opérateurs de condition de correspondance négative, la logique d'évaluation de la valeur de la condition est inversée.

L'exemple suivant permet aux utilisateurs de créer des volumes EC2 et d'appliquer des balises à des volumes pendant la création de volume. Le contexte de la demande doit inclure une valeur pour la clé de contexte `aws:RequestTag/project` et la valeur de la clé de contexte `aws:ResourceTag/environment` peut être n'importe quoi, sauf la production.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:CreateVolume",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:::volume/*",
      "Condition": {
        "StringLike": {
          "aws:RequestTag/project": "*"
        }
      }
    },
    {
      "Effect": "Allow",
```

```
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:region:account:*/*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceTag/environment": "production"
        }
      }
    }
  ]
}
```

Le contexte de demande doit inclure une valeur de balise de projet et ne peut pas être créé pour qu'une ressource de production puisse invoquer l'effet `Allow`. Le volume EC2 suivant a été créé avec succès, car le nom du projet est `Feature3` avec une balise de ressource `QA`.

```
aws ec2 create-volume \
    --availability-zone us-east-1a \
    --volume-type gp2 \
    --size 80 \
    --tag-specifications 'ResourceType=volume,Tags=[{Key=project,Value=Feature3},
{Key=environment,Value=QA}]'
```

Exemple : un bloc de conditions avec plusieurs clés et valeurs de contexte à valeur unique

Lorsqu'un bloc de conditions contient plusieurs clés de contexte et que chaque clé de contexte possède plusieurs valeurs, chaque clé de contexte doit aboutir à true pour au moins une valeur de clé afin que l'effet `Allow` ou `Deny` souhaité soit invoqué. Lorsque vous utilisez des opérateurs de condition de correspondance négative, la logique d'évaluation de la valeur de clé de contexte est inversée.

L'exemple suivant permet aux utilisateurs de démarrer et d'exécuter des tâches sur des clusters Amazon Elastic Container Service.

- Le contexte de la demande doit inclure `production` OR `pre-prod` pour la clé de contexte `aws:RequestTag/environment`AND.
- La clé de contexte `ecs:cluster` garantit que les tâches sont exécutées sur les clusters ECS d'ARN `default1` OR `default2`.

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:RunTask",
        "ecs:StartTask"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/environment": [
            "production",
            "prod-backup"
          ]
        },
        "ArnEquals": {
          "ecs:cluster": [
            "arn:aws:ecs:us-east-1:111122223333:cluster/default1",
            "arn:aws:ecs:us-east-1:111122223333:cluster/default2"
          ]
        }
      }
    }
  ]
}
```

Éléments des politiques IAM : variables et balises

Utilisez des variables de stratégie AWS Identity and Access Management (IAM) comme espaces réservés lorsque vous ne connaissez pas la valeur exacte d'une clé de ressource ou de condition lorsque vous rédigez la politique.

> **Note**
>
> Si vous AWS ne parvenez pas à résoudre une variable, l'intégralité de l'instruction peut être invalide. Par exemple, si vous utilisez la variable `aws:TokenIssueTime`, elle est résolue en une valeur uniquement lorsque le demandeur s'est authentifié à l'aide des informations d'identification temporaires (un rôle IAM). Pour empêcher les variables de provoquer des instructions non valides, utilisez le... IfExists opérateur de condition.

Rubriques

- Introduction
- Utilisation de variables dans les politiques
- Les balises comme variables de la politique
- Éléments dans lesquels vous pouvez utiliser des variables de politique
- Variables de politique sans valeur
- Informations de demande que vous pouvez utiliser dans les variables de politique
- Spécification des valeurs par défaut
- Pour plus d'informations

Introduction

Dans les politiques IAM, de nombreuses actions vous permettent de fournir un nom pour des ressources spécifiques auxquelles vous voulez contrôler l'accès. Par exemple, la politique suivante permet aux utilisateurs d'afficher, de lire et d'écrire les objets dans le compartiment S3 `DOC-EXAMPLE-BUCKET` pour les projets `marketing`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
      "Condition": {"StringLike": {"s3:prefix": ["marketing/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/marketing/*"]
    }
  ]
}
```

Dans certains cas, vous ne connaissez pas le nom exact de la ressource lorsque vous écrivez la politique. Vous souhaiterez peut-être généraliser la politique de façon à ce qu'elle fonctionne pour de nombreux utilisateurs sans avoir à faire une copie unique de celle-ci pour chaque utilisateur. Au lieu de créer une politique distincte pour chaque utilisateur, nous vous recommandons de créer une politique de groupe unique qui s'applique à tous les utilisateurs du groupe.

Utilisation de variables dans les politiques

Vous pouvez définir des valeurs dynamiques dans des politiques en utilisant des variables de politique qui définissent des espaces réservés dans une politique.

Les variables sont marquées à l'aide d'un préfixe **$** suivi d'une paire d'accolades (**{ }**) qui incluent le nom de variable de la valeur issue de la demande.

Lorsque la politique est évaluée, les variables de politique sont remplacées par les valeurs provenant des clés contextuelles conditionnelles passées dans la requête. Les variables peuvent être utilisées dans les politiques basées sur l'identité, les politiques de ressources, les politiques de contrôle des services, les politiques de session et les politiques de point de terminaison d'un VPC. Les politiques basées sur l'identité utilisées comme limites d'autorisations prennent également en charge les variables de politique.

Les clés de contexte des conditions globales peuvent être utilisées comme variables dans les demandes entre les AWS services. Les clés de condition spécifiques à un service peuvent également être utilisées comme variables lors de l'interaction avec des ressources AWS , mais elles ne sont disponibles que lorsque des requêtes sont effectuées auprès de ressources qui les prennent en charge. Pour obtenir la liste des clés contextuelles disponibles pour chaque AWS service et ressource, consultez la référence d'autorisation des services. Dans certains cas, vous ne pouvez pas renseigner les clés contextuelles des conditions globales avec une valeur. Pour en savoir plus sur chaque clé, consultez AWS Clés de contexte de condition globale .

⚠ Important

- Les noms des clés ne sont pas sensibles à la casse. Par exemple, `aws:CurrentTime` équivaut à `AWS:currenttime`.
- Vous pouvez utiliser n'importe quelle clé de condition à valeur unique comme variable. Vous ne pouvez pas utiliser de clé de condition à valeurs multiples en tant que variable.

L'exemple suivant illustre une politique pour un rôle ou un utilisateur IAM qui remplace un nom de ressource spécifique par une variable de politique. Vous pouvez réutiliser cette politique en tirant parti de la clé de condition `aws:PrincipalTag`. Lorsque cette politique est évaluée, `${aws:PrincipalTag/team}` autorise les actions uniquement si le nom du compartiment se termine par un nom d'équipe issu de la balise de principal `team`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
      "Condition": {"StringLike": {"s3:prefix": ["${aws:PrincipalTag/team}/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/${aws:PrincipalTag/team}/*"]
    }
  ]
}
```

La variable est marquée à l'aide d'un préfixe $ suivi d'une paire d'accolades (`{ }`). À l'intérieur des caractères `${ }`, vous pouvez inclure le nom de la valeur de la demande que vous souhaitez utiliser dans la politique. Les valeurs disponibles sont détaillées plus loin sur cette page.

Pour obtenir la liste détaillé de cette clé de condition globale, consultez aws :PrincipalTag//tag-key dans la liste des clés de condition globales.

> **Note**
>
> Pour utiliser des variables de politique, vous devez inclure l'élément `Version` dans une instruction et la version doit être définie sur une version prenant en charge de telles variables. Les variables ont été introduites dans la version `2012-10-17`. Les versions antérieures du langage de politique ne prennent pas en charge les variables de politique. Si vous n'incluez pas l'élément `Version` et que la valeur correspond à une date de version appropriée, les

variables telles que `${aws:username}` sont traitées comme des chaînes littérales dans la politique.

Un élément de politique `Version` varie d'une version de politique. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. En revanche, une version de politique est créée lorsque vous apportez des modifications à une politique gérée par le client dans IAM. La politique modifiée ne remplace pas la politique existante. À la place, IAM crée une nouvelle version de la politique gérée. Pour en savoir plus sur l'élément de politique `Version`, consultez the section called “Version”. Pour en savoir plus sur les versions de politiques, consultez the section called “Gestion des versions des politiques IAM”.

Une politique qui permet à un principal d'obtenir des objets à partir du chemin /David d'un compartiment S3 ressemble à ceci :

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["arn:aws:s3::DOC-EXAMPLE-BUCKET/David/*"]
  }]
}
```

Si cette politique est attaché à l'utilisateur `David`, celui-ci obtient les objets de son propre compartiment S3, mais il vous faudra créer une politique distincte pour chaque utilisateur incluant le nom utilisateur. Vous devez ensuite attacher chaque politique aux utilisateurs individuels.

À l'aide d'une variable de politique, vous pouvez créer des politiques réutilisables. La politique suivante permet à un utilisateur d'obtenir des objets à partir d'un compartiment Amazon S3 si la valeur de la clé de balise `aws:PrincipalTag` correspond à la valeur de la clé de balise `owner` transmise dans la demande.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowUnlessOwnedBySomeoneElse",
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
```

```
    "Resource": ["*"],
    "Condition": {
        "StringEquals": {
          "s3:ExistingObjectTag/owner": "${aws:PrincipalTag/owner}"
        }
      }
    }
  ]
}
```

Lorsque vous utilisez une variable de politique à la place d'un utilisateur, il n'est pas nécessaire d'avoir une politique distincte pour chaque utilisateur. Dans l'exemple suivant, la politique est attachée à un rôle IAM endossé par les Product Managers à l'aide d'informations d'identification de sécurité temporaires. Lorsqu'un utilisateur demande à ajouter un objet Amazon S3, IAM remplace la valeur de balise `dept` de la demande actuelle pour la variable `${aws:PrincipalTag}`, puis évalue la politique.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowOnlyDeptS3Prefix",
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/${aws:PrincipalTag/dept}/*"],
    }
  ]
}
```

Les balises comme variables de la politique

Dans certains AWS services, vous pouvez associer vos propres attributs personnalisés aux ressources créées par ces services. Par exemple, vous pouvez appliquer des balises aux compartiments Amazon S3 ou aux utilisateurs IAM. Ces balises sont des paires clé-valeur. Vous définissez le nom de clé de balise et la valeur qui est associée à ce nom de clé. Par exemple, vous pouvez créer une balise avec une clé **`department`** et une valeur **`Human Resources`**. Pour plus d'informations sur le balisage des entités IAM, consultez Balisage des ressources IAM. Pour plus d'informations sur le balisage des ressources créées par d'autres services AWS , reportez-vous à la documentation de ce service. Pour plus d'informations sur l'utilisation de Tag Editor, consultez Utilisation de Tag Editor dans le Guide de l'utilisateur AWS Management Console .

Vous pouvez baliser des ressources IAM pour simplifier la découverte, l'organisation et le suivi de vos ressources IAM. Vous pouvez aussi baliser les identités IAM pour contrôler l'accès aux ressources ou au balisage lui-même. Pour en savoir plus sur l'utilisation des balises pour contrôler l'accès, consultez Contrôle de l'accès aux et pour les utilisateurs et rôles IAM à l'aide de balises.

Éléments dans lesquels vous pouvez utiliser des variables de politique

Vous pouvez utiliser des variables de politique dans l'élément `Resource` et les comparaisons de chaîne de l'élément `Condition`.

Élément de ressource

Vous pouvez utiliser une variable de politique dans l'élément `Resource`, mais uniquement dans la partie ressource de l'ARN. Cette partie de l'ARN apparaît après le cinquième deux-points (:). Vous ne pouvez pas utiliser une variable pour remplacer des parties de l'ARN avant le cinquième deux-points, par exemple le service ou le compte. Pour de plus amples informations sur le format ARN, veuillez consulter ARN IAM.

Pour remplacer une partie d'un ARN avec une valeur de balise, encadrez le préfixe et le nom de clé avec `${ }`. Par exemple, l'élément de ressource suivant fait référence à un seul compartiment qui est nommé à l'identique de la valeur de la balise du service de l'utilisateur demandeur.

```
"Resource": ["arn:aws::s3:::bucket/${aws:PrincipalTag/department}"]
```

De nombreuses AWS ressources utilisent des ARN contenant un nom créé par l'utilisateur. La politique IAM suivante garantit que seuls les utilisateurs concernés dont les valeurs de balises access-project, access-application et access-environment sont correspondantes peuvent modifier leurs ressources. En outre, en utilisant le caractère générique *, ils peuvent autoriser des suffixes de nom de ressources personnalisés.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessBasedOnArnMatching",
      "Effect": "Allow",
      "Action": [
        "sns:CreateTopic",
        "sns:DeleteTopic"],
      "Resource": ["arn:aws:sns:*:*:${aws:PrincipalTag/access-project}-
${aws:PrincipalTag/access-application}-${aws:PrincipalTag/access-environment}-*"
```

```
      ]
    }
  ]
}
```

Élément de condition

Vous pouvez utiliser une variable de politique pour les valeurs `Condition` dans n'importe quelle condition impliquant les opérateurs de chaîne ou les opérateurs ARN. Les opérateurs de chaîne incluent `StringEquals`, `StringLike` et `StringNotLike`. Les opérateurs ARN incluent `ArnEquals` et `ArnLike`. Vous ne pouvez pas utiliser une variable de politique avec d'autres opérateurs, tels que les opérateurs `Numeric`, `Date`, `Boolean`, `Binary`, `IP Address` ou `Null`. Pour de plus amples informations sur les opérateurs de condition, veuillez consulter Éléments de politique JSON IAM : Opérateurs de condition.

Lorsque vous faites référence à une balise dans une expression de l'élément `Condition`, utilisez le préfixe et le nom de clé pertinents comme clé de condition. Utilisez ensuite la valeur que vous souhaitez tester dans la valeur de condition.

Par exemple, l'exemple de politique suivant autorise l'accès complet aux utilisateurs, mais uniquement si la balise `costCenter` est attachée à l'utilisateur. La balise doit également avoir la valeur `12345` ou `67890`. Si la balise n'a pas de valeur, ou toute autre valeur, la demande échoue.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
          "iam:*user*"
       ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:ResourceTag/costCenter": [ "12345", "67890" ]
        }
      }
    }
  ]
}
```

Variables de politique sans valeur

Lorsque les variables de politique font référence à une clé contextuelle de condition qui n'a aucune valeur ou qui n'est pas présente dans le contexte d'autorisation d'une requête, la valeur est effectivement nulle. Il n'existe pas de valeur égale ou similaire. Les clés contextuelles de condition peuvent ne pas être présentes dans le contexte d'autorisation lorsque :

- Vous utilisez des clés contextuelles de condition spécifiques au service dans les requêtes adressées à des ressources qui ne prennent pas en charge cette clé de condition.
- Les balises sur les principaux, les sessions, les ressources ou les demandes IAM ne sont pas présentes.
- Les autres cas répertoriés pour chaque condition globale sont incluses dans le contexte AWS clés contextuelles de condition globale.

Lorsque vous utilisez une variable sans valeur dans l'élément de condition d'une politique IAM, Éléments de politique JSON IAM : Opérateurs de condition comme `StringEquals` ou `StringLike` ne correspondent pas et la déclaration de politique ne prend pas effet.

Les opérateurs de condition inversés comme `StringNotEquals` ou `StringNotLike` correspondent à une valeur nulle, car la valeur de la clé conditionnelle par rapport à laquelle ils testent n'est pas égale ou similaire à la valeur effectivement nulle.

Dans l'exemple suivant, `aws:principaltag/Team` doit être égal à `s3:ExistingObjectTag/Team` pour autoriser l'accès. L'accès est explicitement refusé lorsque `aws:principaltag/Team` n'est pas défini. Si une variable qui n'a aucune valeur dans le contexte d'autorisation est utilisée dans le cadre de l'élément `Resource` or `NotResource` d'une politique, la ressource qui inclut une variable de politique sans valeur ne correspondra à aucune ressource.

```
{
  "Version": "2012-10-17",
  "Statement": [
   {
    "Effect": "Deny",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::/example-bucket/*",
    "Condition": {
      "StringNotEquals": {
        "s3:ExistingObjectTag/Team": "${aws:PrincipalTag/Team}"
       }
      }
```

```
    }
  ]
}
```

Informations de demande que vous pouvez utiliser dans les variables de politique

Vous pouvez utiliser l'élément `Condition` d'une politique JSON pour comparer des clés dans le contexte de demande avec les valeurs de clé spécifiées dans votre politique. Lorsque vous utilisez une variable de politique, AWS substitue une valeur de la clé de contexte de demande à la place de la variable dans votre politique.

Valeurs de la clé du principal

Les valeurs de `aws:username`, `aws:userid` et `aws:PrincipalType` dépendent du type de principal ayant initialisé la demande. Par exemple, la demande peut être effectuée à l'aide des informations d'identification d'un utilisateur IAM, d'un rôle IAM ou de l' Utilisateur racine d'un compte AWS. La liste de suivante affiche les valeurs de ces clés pour différents types de principaux.

- Utilisateur racine d'un compte AWS
  - `aws:username` : (absent)
  - `aws:userid`: Compte AWS ID
  - `aws:PrincipalType`: `Account`
- Utilisateur IAM
  - `aws:username`: *`nom-utilisateur-IAM`*
  - `aws:userid`: ID unique
  - `aws:PrincipalType`: `User`
- Utilisateur fédéré
  - `aws:username` : (absent)
  - `aws:userid`: *`account`*:*`caller-specified-name`*
  - `aws:PrincipalType`: `FederatedUser`
- Utilisateur fédéré web et utilisateur fédéré SAML

> **Note**
>
> Pour plus d'informations sur les clés de politique disponibles lorsque vous utilisez la fédération OIDC, consultez???.

  - `aws:username` : (absent)
  - `aws:userid` : (absent)
  - `aws:PrincipalType`: `AssumedRole`
- Rôle endossé
  - `aws:username` : (absent)
  - `aws:userid`: *`role-id`*:*`caller-specified-role-name`*
  - `aws:PrincipalType`: `Assumed role`
- Rôle affecté à une instance Amazon EC2
  - `aws:username` : (absent)
  - `aws:userid`: *`role-id`*:*`ec2-instance-id`*
  - `aws:PrincipalType`: `Assumed role`
- Appelant anonyme (Amazon SQS, Amazon SNS et Amazon S3 uniquement)
  - `aws:username` : (absent)
  - `aws:userid` : (absent)
  - `aws:PrincipalType`: `Anonymous`

Pour les éléments de cette liste, notez les points suivants :

- absent signifie que la valeur ne figure pas dans les informations de la demande en cours et par conséquent, toute tentative de correspondance échoue et rend l'instruction non valide.
- *`id-rôle`* est un identifiant unique attribué à chaque rôle lors de sa création. Vous pouvez afficher l'ID du rôle à l'aide de la AWS CLI commande suivante : `aws iam get-role --role-name` *`rolename`*
- *`nom-spécifié-par-principal`* et *`nom-rôle-spécifié-par-principal`* sont des noms qui sont transmis par le processus appelant (comme une application ou un service) lorsqu'il effectue un appel pour obtenir des informations d'identification temporaires.
- *`ec2-instance-id`* est une valeur affectée à l'instance lors de son lancement ; elle s'affiche sur la page Instances de la console Amazon EC2. Vous pouvez également afficher l'ID de l'instance en exécutant la AWS CLI commande suivante : `aws ec2 describe-instances`

Informations relatives aux utilisateurs fédérés disponibles dans les demandes

Les utilisateurs fédérés sont des utilisateurs authentifiés à l'aide d'un système autre qu'IAM. Par exemple, une entreprise peut avoir une application à utiliser en interne qui passe des appels à AWS. L'attribution d'une identité IAM à chaque utilisateur de l'entreprise qui utilise l'application peut ne pas convenir. Au lieu de cela, l'entreprise peut utiliser une application proxy (intermédiaire) qui dispose d'une identité IAM unique ou un fournisseur d'identité (IdP) SAML. L'application proxy ou l'IdP SAML authentifie les utilisateurs individuels via le réseau d'entreprise. Une application proxy peut ensuite utiliser son identité IAM pour obtenir des informations d'identification de sécurité temporaires pour des utilisateurs individuels. Un IdP SAML peut en effet échanger des informations d'identité contre des informations AWS de sécurité temporaires. Les informations d'identification temporaires peuvent ensuite être utilisées pour accéder aux AWS ressources.

De même, vous pouvez créer une application pour appareil mobile qui doit accéder aux ressources AWS . Dans ce cas, vous pouvez utiliser la fédération OIDC, où l'application authentifie l'utilisateur à l'aide d'un fournisseur d'identité connu tel que Login with Amazon, Amazon Cognito, Facebook ou Google. L'application peut ensuite utiliser les informations d'authentification de l'utilisateur à partir de ces fournisseurs pour obtenir les informations d'identification de sécurité temporaires permettant d'accéder aux ressources AWS .

La méthode recommandée pour utiliser la fédération OIDC consiste à tirer parti d'Amazon Cognito et AWS des SDK mobiles. Pour plus d’informations, consultez les ressources suivantes :

- Guide de l'utilisateur d'Amazon Cognito
- Scénarios courants d'informations d'identification temporaires

Caractères spéciaux

Quelques variables de politique prédéfinies spéciales sont dotées de valeurs fixes qui vous permettent de représenter des caractères qui, sinon, ont une signification spéciale. Si ces caractères spéciaux font partie de la chaîne que vous essayez de faire correspondre et qu'ils sont insérés littéralement, ils sont susceptibles d'être mal interprétés. Par exemple, un astérisque (*) inséré dans la chaîne sera interprété comme un caractère générique, ce qui correspond à n'importe quel caractère, au lieu d'un caractère * littéral. Dans ces cas, vous pouvez utiliser les variables de politique prédéfinies suivantes :

- ${*} – permet d'obtenir le caractère * (astérisque).
- ${?} – permet d'obtenir le caractère ? (point d'interrogation).

- ${$} – permet d'obtenir le caractère $ (dollar).

Ces variables de politique prédéfinies peuvent être utilisées dans n'importe quelle chaîne acceptant les variables de politique standard.

Spécification des valeurs par défaut

Pour ajouter une valeur par défaut à une variable, entourez la valeur par défaut de guillemets simples (`' '`), puis séparez le texte de la variable et la valeur par défaut par une virgule et une espace (`, `).

Par exemple, si un principal est labelisé avec l'interface `team=yellow`, il peut accéder au compartiment Amazon S3 `ExampleCorp's` nommé `DOC-EXAMPLE-BUCKET-yellow`. Une politique avec cette ressource permet aux membres de l'équipe d'accéder à leur compartiment d'équipe, mais pas à ceux des autres équipes. Pour les utilisateurs sans identifications d'équipe, il définit une valeur par défaut de l'interface `company-wide` pour le nom du compartiment. Ces utilisateurs peuvent accéder uniquement au compartiment `DOC-EXAMPLE-BUCKET-company-wide` où ils peuvent afficher des informations générales, telles que des instructions pour rejoindre une équipe.

```
"Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET-${aws:PrincipalTag/team, 'company-wide'}"
```

Pour plus d'informations

Pour plus d'informations sur les politiques, consultez les ressources suivantes :

- Politiques et autorisations dans IAM
- Exemples de politiques basées sur l'identité IAM
- Références des éléments de politique JSON IAM
- Logique d'évaluation de politiques
- Fédération OIDC

Éléments de politique JSON IAM : Types de données pris en charge

Cette section répertorie les types de données pris en charge lors de la spécification de valeurs dans des politiques JSON. Le langage de politique ne prend pas en charge tous les types d'éléments de politique ; reportez-vous aux sections précédentes pour plus d'informations sur chaque élément.

- Chaînes

- Nombres (Ints et Floats)
- Booléen
- Null
- Listes
- Mappages
- Structs (simples mappages intégrés)

Le tableau suivant mappe chaque type de données à la sérialisation. Notez que toutes les politiques doivent utiliser le format UTF-8. Pour plus d'informations sur les types de données JSON, reportez-vous à RFC 4627.

Type	JSON
Chaîne	Chaîne
Entier	Nombre
Float	Nombre
Booléen	true false
Null	null
Date	Chaîne conforme au profil W3C d'ISO 8601
IpAddress	Chaîne conforme à RFC 4632
Liste	Tableau
Objet	Objet

Logique d'évaluation de politiques

Lorsqu'un principal essaie d'utiliser l' AWS Management Console AWS API, ou le AWS CLI, ce principal envoie une demande à AWS. Lorsqu'un AWS service reçoit la demande, AWS effectue plusieurs étapes pour déterminer s'il convient d'autoriser ou de refuser la demande.

1. Authentification : authentifie d' AWS abord le principal qui fait la demande, si nécessaire. Cette étape n'est pas nécessaire pour quelques services, tels qu'Amazon S3, qui autorisent certaines demandes provenant d'utilisateurs anonymes.
2. Traitement du contexte de la demande— AWS traite les informations recueillies dans la demande afin de déterminer quelles politiques s'appliquent à la demande.
3. Évaluation des politiques dans un compte unique— AWS évalue tous les types de politiques, ce qui influe sur l'ordre dans lequel les politiques sont évaluées.
4. Identification d'une demande autorisée ou refusée dans un compte— traite AWS ensuite les politiques en fonction du contexte de la demande pour déterminer si la demande est autorisée ou refusée.

Traitement du contexte de la demande

AWS traite la demande pour recueillir les informations suivantes dans un contexte de demande :

- Actions (or operations) (Actions [ou opérations]) : les actions ou opérations que le principal souhaite exécuter.
- Ressources : objet de AWS ressource sur lequel les actions ou opérations sont effectuées.
- Principal : l'utilisateur, le rôle, l'utilisateur fédéré ou l'application à l'origine de la demande. Les informations sur le principal incluent les politiques associées à ce dernier.
- Données d'environnement : informations sur l'adresse IP, l'agent utilisateur, le statut SSL ou le moment de la journée.
- Données de ressources : données liées à la ressource qui est demandée. Par exemple, ces informations peuvent inclure le nom d'une table DynamoDB ou une balise sur une instance Amazon EC2.

AWS utilise ensuite ces informations pour rechercher les politiques qui s'appliquent au contexte de la demande.

Évaluation des politiques dans un compte unique

La manière dont AWS les politiques sont évaluées dépend des types de politiques qui s'appliquent au contexte de la demande. Les types de politique suivants sont répertoriés par ordre de fréquence et utilisables dans un seul Compte AWS. Pour plus d'informations sur ces types de politique, consultez Politiques et autorisations dans IAM. Pour savoir comment AWS évalue les politiques d'accès entre comptes, voir. Logique d'évaluation des politiques entre comptes

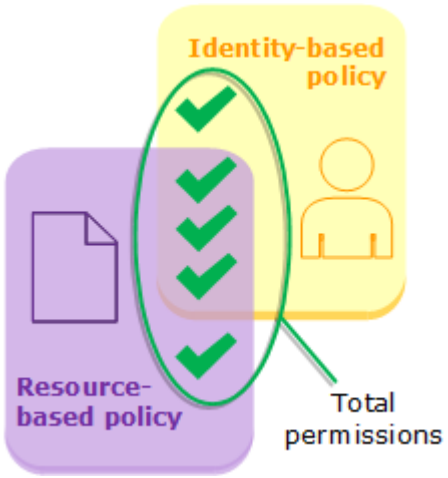
1. Politiques basées sur l'identité : les politiques basées sur l'identité sont attachées à une identité IAM (utilisateur, groupe d'utilisateurs ou rôle) et octroient des autorisations aux entités IAM (utilisateurs et rôles). Si seules les politiques basées sur l'identité s'appliquent à une demande, alors AWS vérifie toutes ces politiques pour au moins une d'entre elles. `Allow`
2. Politiques basées sur les ressources – Les politiques basées sur les ressources accordent des autorisations au principal (compte, utilisateur, rôle et principaux de session tels que les sessions de rôle et les utilisateurs fédérés IAM) spécifié comme principal. Les autorisations définissent ce que le principal peut faire avec la ressource à laquelle la politique est attachée. Si les politiques basées sur les ressources et les politiques basées sur l'identité s'appliquent toutes deux à une demande, alors AWS vérifie toutes les politiques pour au moins une d'entre elles. `Allow` Lorsque les politiques basées sur les ressources sont évaluées, l'ARN principal qui est spécifié dans la politique détermine si les rejets implicites dans d'autres types de politiques sont applicables à la décision finale.
3. IAM permissions boundaries (Limites d'autorisations) : les limites d'autorisations sont une fonctionnalité avancée qui définit les autorisations maximales qu'une politique basée sur les identités peut accorder à une entité IAM (utilisateur ou rôle). Lorsque vous définissez une limite d'autorisations pour une entité, l'entité peut effectuer uniquement les actions autorisées par ses deux ses stratégies basées sur l'identité et ses limites d'autorisations. Dans certains cas, un refus implicite dans une limite d'autorisations peut limiter les autorisations accordées par une politique basée sur les ressources. Pour en savoir plus, veuillez consulter Identification d'une demande autorisée ou refusée dans un compte plus loin dans cette rubrique.
4. AWS Organizations politiques de contrôle des services (SCP) — Organisations Les SCP spécifient les autorisations maximales pour une organisation ou une unité organisationnelle (UO). Le SCP maximum s'applique aux principaux des comptes des membres, y compris à chacun d'entre eux. Utilisateur racine d'un compte AWS Si une stratégie de contrôle de service est présente, les politiques basées sur l'identité et sur les ressources accordent des autorisations aux principaux dans les comptes membres uniquement si ces politiques et la stratégie de contrôle de service autorise l'action. Si une limite d'autorisations et une politique de contrôle de service sont présentes, la limite, la politique de contrôle de service et la politique basée sur l'identité doivent toutes autoriser l'action.
5. Session policies (Politiques de session) : les politiques de session sont des politiques avancées que vous transmettez en tant que paramètres lorsque vous créez par programmation une session temporaire pour un rôle ou un utilisateur fédéré. Pour créer une session de rôle par programmation, utilisez l'une des opérations d'API `AssumeRole*`. Lorsque vous procédez ainsi et transmettez de politiques de session, les autorisations de session obtenues sont une combinaison de la politique basée sur l'identité de l'entité IAM et des politiques de session. Pour créer une

session d'utilisateur fédéré, vous utilisez des clés d'accès d'utilisateur IAM pour appeler par programmation l'opération d'API `GetFederationToken`. Une politique basée sur les ressources a un autre effet sur l'évaluation des autorisations de la politique de session. La différence dépend de si l'utilisateur ou l'ARN du rôle ou de la session de l'ARN est répertorié en tant que principal dans la politique basée sur les ressources. Pour plus d'informations, veuillez consulter Politiques de session.

Pour rappel, un refus explicite dans l'une de ces politiques remplace l'autorisation.

Évaluation des politiques basées sur l'identité avec des politiques basées sur des ressources

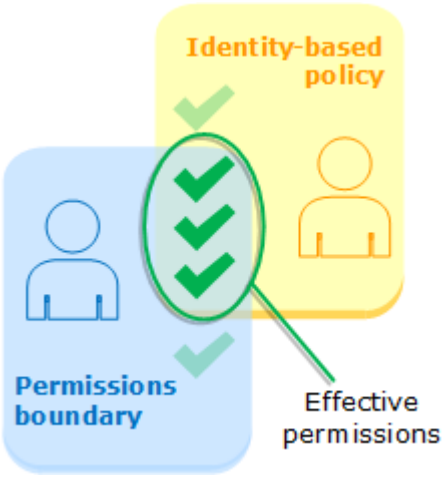
Les politiques basées sur l'identité et sur les ressources accordent des autorisations aux identités ou ressources auxquelles elles sont associées. Lorsqu'une entité IAM (utilisateur ou rôle) demande l'accès à une ressource au sein du même compte, AWS elle évalue toutes les autorisations accordées par les politiques basées sur l'identité et les ressources. Les autorisations obtenues constituent le total des autorisations des deux types. Si une action est autorisée par une stratégie basée sur l'identité, une politique basée sur les ressources, ou les deux, l'action est autorisée. AWS Un refus explicite dans l'une ou l'autre de ces stratégies remplace l'autorisation.



Évaluation des politiques basées sur l'identité avec des limites d'autorisations

Lors de l' AWS évaluation des politiques basées sur l'identité et des limites d'autorisations pour un utilisateur, les autorisations qui en résultent sont à l'intersection des deux catégories. En d'autres termes, lorsque vous ajoutez une limite d'autorisations à un utilisateur avec les politiques basées sur l'identité existantes, vous pouvez réduire les actions exécutées par l'utilisateur. Sinon, lorsque vous supprimez une limite d'autorisations à partir d'un utilisateur, vous pouvez augmenter les actions qu'il peut exécuter. Un refus explicite dans l'une ou l'autre de ces stratégies remplace l'autorisation. Pour

afficher des informations sur la façon dont les autres types de politique sont évalués avec des limites d'autorisations, consultez Évaluation des autorisations effectives avec limites.



Évaluation des politiques basées sur l'identité avec des SCP Organizations

Lorsqu'un utilisateur appartient à un compte qui est membre d'une organisation, les autorisations obtenues sont une combinaison des politiques de l'utilisateur et de la SCP. Cela signifie qu'une action doit être autorisée par la politique basée sur l'identité et la SCP. Un refus explicite dans l'une ou l'autre de ces stratégies remplace l'autorisation.

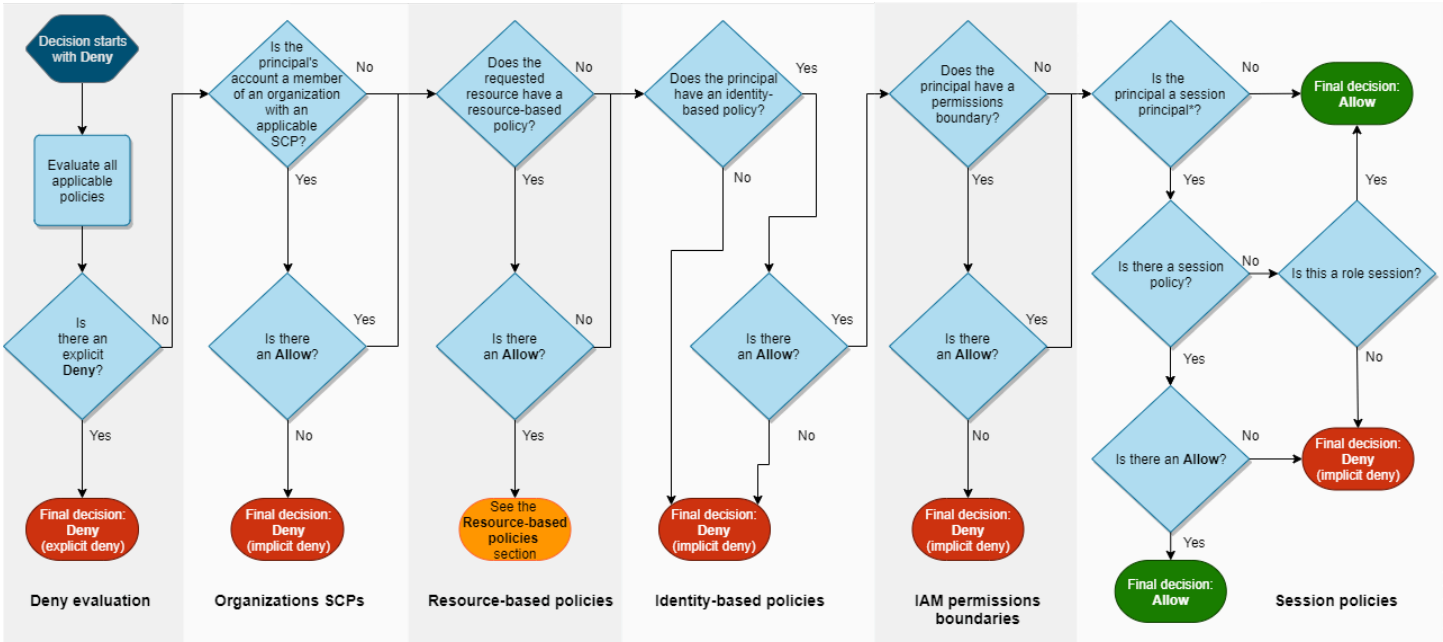
Vous pouvez savoir si votre compte est membre d'une organisation dans AWS Organizations. Les membres de l'organisation pourraient être affectés par une SCP. Pour afficher ces données à l'aide de la AWS CLI commande ou de AWS l'opération d'API, vous devez disposer des autorisations nécessaires à l'`organizations:DescribeOrganization`action pour votre entité Organizations. Vous devez disposer des autorisations supplémentaires pour effectuer l'opération dans la console Organizations. Pour savoir si un SCP refuse l'accès à une demande spécifique ou pour modifier vos autorisations effectives, contactez votre AWS Organizations administrateur.

Identification d'une demande autorisée ou refusée dans un compte

Supposons qu'un principal envoie une demande AWS d'accès à une ressource dans le même compte que l'entité du principal. Le code d' AWS exécution décide si la demande doit être acceptée ou refusée. AWS évalue toutes les politiques applicables au contexte de la demande. Ce qui suit est un résumé de la logique AWS d'évaluation des politiques au sein d'un seul compte.

- Par défaut, toutes les demandes sont implicitement refusées, à l'exception de Utilisateur racine d'un compte AWS, qui dispose d'un accès complet.
- Une autorisation explicite dans une politique basée sur l'identité ou les ressources remplace cette valeur par défaut.
- Si une limite d'autorisations, SCP Organizations ou politique de session est présente, elle peut remplacer l'autorisation avec un refus implicite.
- Un refus explicite dans n'importe quelle politique remplace toutes les autorisations.

Le diagramme suivant décrit comment la décision est prise. Cet organigramme ne couvre pas l'impact des politiques basées sur les ressources et des rejets implicites dans d'autres types de politiques.



*A session principal is either a role session or an IAM federated user session.

1. Deny evaluation (Refuser l'évaluation) : par défaut, toutes les demandes sont refusées. Il s'agit d'un refus implicite. Le code d' AWS application évalue toutes les politiques du compte qui s'appliquent à la demande. Il s'agit notamment des AWS Organizations SCP, des politiques

basées sur les ressources, des politiques basées sur l'identité, des limites d'autorisations IAM et des politiques de session. Dans toutes ces politiques, le code d'application recherche une instruction `Deny` (Refuser) qui s'applique à la demande. Ceci est appelé un refus explicite. Si le code trouve un refus explicite qui s'applique, il retourne la décision finale Deny (Refuser). S'il n'y a pas de refus explicite, l'évaluation du code d'application se poursuit.

2. Organizations SCP — Le code d'application évalue ensuite les politiques de contrôle des AWS Organizations services (SCP) qui s'appliquent à la demande. Les stratégies de contrôle de service s'appliquent aux principaux du compte auquel elles sont rattachées. Si le code d'application ne trouve aucune instruction `Allow` applicable dans les SPC, la demande est explicitement refusée, même si le refus est implicite. Le code d'application retourne la décision finale Deny (Refuser). S'il n'y a pas de SCP, ou si la SCP autorise l'action demandée, l'évaluation du code d'exécution se poursuit.
3. Politiques basées sur les ressources – Au sein d'un même compte, les politiques basées sur les ressources influencent l'évaluation des politiques différemment selon le type de principal accédant à la ressource et le principal autorisé dans la politique basée sur les ressources. Selon le type de principal, un `Allow` dans une politique basée sur les ressources peut aboutir à une décision finale de `Allow`, même si un rejet implicite dans une politique basée sur une identité, une limite d'autorisations ou une politique de séance est présente.

   Pour la plupart des ressources, vous n'avez besoin d'une autorisation explicite pour le principal que dans une politique basée sur l'identité ou dans une politique basée sur les ressources pour accorder l'accès. Les politiques de confiance des rôles IAM et politiques de clé KMS sont des exceptions à cette logique, car ils doivent explicitement autoriser l'accès pour les principaux.

   La logique des politiques basées sur les ressources diffère des autres types de politiques si le principal spécifié est un utilisateur IAM, un rôle IAM ou un principal de séance. Les principaux de séance incluent les séances de rôle IAM ou une séance d'utilisateur fédérée IAM. Si une politique basée sur les ressources accorde des autorisations directement à l'utilisateur IAM ou au principal de séance qui fait la demande, un rejet implicite dans une politique basée sur l'identité, une limite d'autorisations ou une politique de séance n'a pas d'incidence sur la décision finale.

   Le tableau suivant vous aide à comprendre l'impact des politiques basées sur les ressources pour différents types de principal lorsque des rejets implicites sont présents dans les politiques basées sur une l'identité, les limites d'autorisations et les politiques de séance.

Politiques basées sur les ressources et rejets implicites dans d'autres types de politique (même compte)

Principal faisant la demande	Politique basée sur une ressource	Politique basée sur l'identité	Limite d'autorisations	Politique de séance	Résultat	Raison
Rôle IAM	Ne s’applique pas	Ne s’applique pas	Ne s’applique pas	Ne s’applique pas	Ne s’applique pas	Un rôle lui-même ne peut pas faire de demande. Les demandes sont faites avec la séance de rôle une fois qu'un rôle est endossé.

Principal faisant la demande	Politique basée sur une ressource	Politique basée sur l'identité	Limite d'autorisations	Politique de séance	Résultat	Raison
Séance de rôle IAM	Autorise l'ARN du rôle	Refus implicite	Refus implicite	Refus implicite	REJETER	La limite des autorisations et la politique de séance sont évaluées dans le cadre de la décision finale. Un rejet implicite dans l'une ou l'autre des politiques entraîne une décision de REJET.

Principal faisant la demande	Politique basée sur une ressource	Politique basée sur l'identité	Limite d'autorisations	Politique de séance	Résultat	Raison
Séance de rôle IAM	Autorise l'ARN de séance de rôle	Refus implicite	Refus implicite	Refus implicite	AUTORISER	Les autorisations sont accordées directement à la séance. Les autres types de politiques n'affectent pas la décision.
Utilisateur IAM	Autorise l'ARN de l'utilisateur IAM	Refus implicite	Refus implicite	Ne s’applique pas	AUTORISER	Les autorisations sont accordées directement à l'utilisateur. Les autres types de politiques n'affectent pas la décision.

Principal faisant la demande	Politique basée sur une ressource	Politique basée sur l'identité	Limite d'autorisations	Politique de séance	Résultat	Raison
Utilisateur fédéré IAM (`GetFederationToken`)	Autorise l'ARN de l'utilisateur IAM	Refus implicite	Refus implicite	Refus implicite	REJETER	Un rejet implicite dans la limite des autorisations ou dans la politique de séance entraîne un REJET.
Utilisateur fédéré IAM (`GetFederationToken`)	Autorise l'ARN de séance d'utilisateur fédérée IAM	Refus implicite	Refus implicite	Refus implicite	AUTORISER	Les autorisations sont accordées directement à la séance. Les autres types de politiques n'affectent pas la décision.

Principal faisant la demande	Politique basée sur une ressource	Politique basée sur l'identité	Limite d'autorisations	Politique de séance	Résultat	Raison
utilisateur root	Autorise l'ARN de l'utilisateur racine	Ne s'applique pas	Ne s'applique pas	Ne s'applique pas	AUTORISER	L'utilisateur root a un accès complet et illimité à toutes les ressources de votre Compte AWS. Pour connaître la procédure à suivre pour contrôler l'accès aux utilisateurs root pour les comptes dans AWS Organizations, veuillez consulter la rubrique Politiques de contrôle de service (SCP)

Principal faisant la demande	Politique basée sur une ressource	Politique basée sur l'identité	Limite d'autorisations	Politique de séance	Résultat	Raison
						dans le Guide de l'utilisateur des organisations.
AWS service principal	Autorise un directeur AWS de service	Ne s'applique pas	Ne s'applique pas	Ne s'applique pas	AUTORISER	Lorsqu'une politique basée sur les ressources accorde directement des autorisations à un principal de service AWS, les autres types de politiques n'affectent pas la décision.

- Rôle IAM – Les politiques basées sur les ressources qui accordent des autorisations à un ARN de rôle IAM sont limitées par un rejet implicite dans une limite d'autorisations ou une politique de séance.

  Exemple d'ARN de rôle

```
arn:aws:iam::111122223333:role/examplerole
```

- Rôle IAM – Au sein du même compte, les politiques basées sur les ressources qui accordent des autorisations à un ARN de séance de rôle IAM accordent des autorisations directement à la séance de rôle supposée. Les autorisations accordées directement à une séance ne sont pas limitées par un rejet implicite dans une politique basée sur l'identité, une limite d'autorisations ou une politique de séance. Lorsque vous endossez un rôle et que vous faites une demande, le principal qui fait la demande est l'ARN de séance de rôle IAM et non l'ARN du rôle lui-même.

  Exemple d'ARN de séance de rôle

  ```
  arn:aws:sts::111122223333:assumed-role/examplerole/examplerolesessionname
  ```

- Utilisateur IAM – Au sein du même compte, les politiques basées sur les ressources qui accordent des autorisations à un ARN d'utilisateur IAM (qui n'est pas une séance d'utilisateur fédérée) ne sont pas limitées par un rejet implicite d'une politique basée sur l'identité ou d'une limite d'autorisations.

  Exemple d'ARN d'utilisateur IAM

  ```
  arn:aws:iam::111122223333:user/exampleuser
  ```

- Séances d'utilisateur fédérées IAM – Une séance d'utilisateur fédérée IAM est une séance créée en appelant `GetFederationToken`. Lorsqu'un utilisateur fédéré fait une demande, le principal qui effectue la demande est l'ARN d'utilisateur fédéré et non l'ARN de l'utilisateur IAM qui a fédéré. Dans le même compte, les politiques basées sur les ressources accordant des autorisations à un ARN d'utilisateur fédéré accordent des autorisations directement à la séance. Les autorisations accordées directement à une séance ne sont pas limitées par un rejet implicite dans une politique basée sur l'identité, une limite d'autorisations ou une politique de séance.

  Toutefois, si une politique basée sur les ressources accorde une autorisation à l'ARN de l'utilisateur IAM qui s'est fédéré, les demandes effectuées par l'utilisateur fédéré pendant la séance sont limitées par un rejet implicite dans une limite d'autorisation ou une politique de séance.

  Exemple d'ARN de séance d'utilisateur fédérée IAM

  ```
  arn:aws:sts::111122223333:federated-user/exampleuser
  ```

4. Identity-based policies (Politiques basées sur l'identité) : le code vérifie ensuite les politiques basées sur l'identité pour le principal. Pour un utilisateur IAM, cela comprend notamment les politiques d'utilisateur et les politiques de groupes auxquels l'utilisateur appartient. Si aucune politique basée sur l'identité ou aucune instruction dans les politiques basées sur l'identité n'autorise l'action demandée, alors la demande est implicitement refusée et le code renvoie une décision finale de refus. Si une instruction dans n'importe quelle politique basée sur l'identité applicable autorise l'action demandée, le code continue.
5. IAM permissions boundaries (limites d'autorisations IAM) – le code vérifie ensuite si l'entité IAM utilisée par le principal possède une limite d'autorisations. Si la politique qui est utilisée pour définir la limite d'autorisations n'autorise pas l'action demandée, la demande est implicitement refusée. Le code retourne la décision finale Deny (Refuser). S'il n'y a pas de limite d'autorisations, ou si la limite d'autorisations autorise l'action demandée, le code continue.
6. Politiques de séance : Le code vérifie ensuite si le principal est un principal de séance. Les principaux de séance incluent une séance de rôle IAM ou une séance d'utilisateur fédérée IAM. Si le principal n'est pas un principal de séance, le code d'application renvoie une décision finale d'autorisation.

   Pour les principaux de séance, le code vérifie si une politique de séance a été transmise dans la demande. Vous pouvez transmettre une politique de session lorsque vous utilisez l' AWS API AWS CLI or pour obtenir des informations d'identification temporaires pour un rôle ou un utilisateur fédéré IAM.

   - Si une politique de session est présente et n'autorise pas l'action demandée, la demande est implicitement refusée. Le code retourne la décision finale Deny (Refuser).
   - S'il n'y a pas de politique de séance, le code vérifie si le principal est une séance de rôle. Si le principal est une séance de rôle, la demande est autorisée. Sinon, la demande est implicitement rejetée et le code renvoie une décision finale de rejet.
   - Si une politique de séance est présente et autorise l'action demandée, le code d'exécution renvoie une décision finale d'autorisation.
7. Erreurs — Si le code d' AWS application rencontre une erreur à un moment quelconque au cours de l'évaluation, il génère une exception et se ferme.

Exemple d'évaluation de politique basée sur l'identité et sur les ressources

Les types de politiques les plus courants sont celles basées sur l'identité et les ressources. Lorsque l'accès à une ressource est demandé, AWS évalue toutes les autorisations accordées par les

politiques pour au moins une autorisation au sein du même compte. Un rejet explicite dans l'une de ces politiques remplace l'autorisation.

> **Important**
>
> Si la politique basée sur les identités ou celle basée sur les ressources dans le même compte autorise la demande et que l'autre ne l'autorise pas, la demande reste autorisée.

Supposons que le nom d'utilisateur de Carlos soit `carlossalazar` et que ce dernier essaie d'enregistrer un fichier dans le compartiment Amazon S3 `carlossalazar-logs`.

Supposons également que la politique suivante soit attachée à l'utilisateur IAM `carlossalazar`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowS3Self",
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::carlossalazar/*",
                "arn:aws:s3:::carlossalazar"
            ]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::*log*"
```

```
        }
    ]
}
```

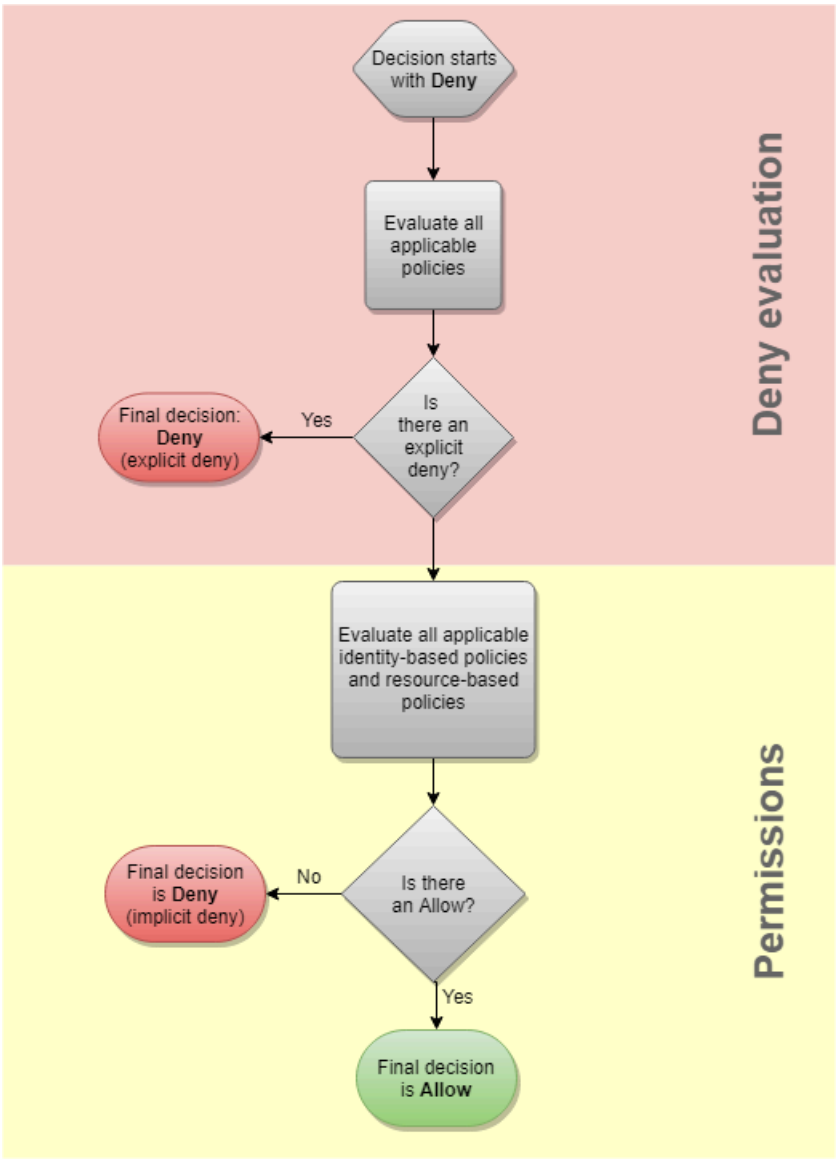
L'instruction `AllowS3ListRead` de cette politique autorise Carlos à afficher une liste de tous les compartiments du compte. L'instruction `AllowS3Self` accorde à Carlos un accès total au compartiment du même nom que son nom d'utilisateur. L'instruction `DenyS3Logs` refuse à Carlos l'accès à n'importe quel compartiment S3 comprenant `log` dans son nom.

De plus, la politique basée sur les ressources suivante (appelée politique de compartiment) est attachée au compartiment `carlossalazar`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/carlossalazar"
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::carlossalazar/*",
                "arn:aws:s3:::carlossalazar"
            ]
        }
    ]
}
```

Cette politique spécifie que seul l'utilisateur `carlossalazar` peut accéder au compartiment `carlossalazar`.

Lorsque Carlos demande l'enregistrement d'un fichier dans le `carlossalazar-logs` bucket, il AWS détermine les politiques qui s'appliquent à la demande. Dans ce cas, seule les politiques basées sur l'identité et les ressources s'appliquent. Il s'agit de deux politiques d'autorisations. La logique d'évaluation est réduite à la logique suivante, car aucune limite d'autorisations ne s'applique.



AWS vérifie d'abord si une `Deny` déclaration s'applique au contexte de la demande. Il en trouve une, car la politique basée sur l'identité refuse explicitement à Carlos l'accès à n'importe quel compartiment S3 utilisé pour la journalisation. Carlos se voit refuser l'accès.

Supposons qu'il se rende compte de son erreur et essaie d'enregistrer le fichier dans le `carlossalazar` bucket. AWS recherche une `Deny` déclaration et n'en trouve aucune. Ensuite, il vérifie les politiques d'autorisations. La politique basée sur l'identité et celle basée sur les ressources autorisent la demande. AWS Autorise donc la demande. Si l'un d'entre eux refuse explicitement l'instruction, alors la demande est refusée. Si l'un des types de politique autorise la demande et que l'autre ne l'autorise pas, la demande reste autorisée.

Différence entre les refus explicites et implicites

Une demande se traduit par un refus explicite si une politique applicable inclut une instruction `Deny` (Refuser). Si les politiques qui s'appliquent à une demande incluent une instruction `Allow` (Autoriser) et une instruction `Deny` (Refuser), alors l'instruction `Deny` a priorité sur l'instruction `Allow` (Autoriser). La demande est refusée explicitement.

Un refus implicite se produit en cas d'absence d'instructions `Deny` (Refuser) et `Allow` (Autoriser) applicables. Étant donné qu'un principal IAM se voit refuser l'accès par défaut, il doit être explicitement autorisé à effectuer une action. Sinon, il se voit refuser l'accès implicitement.

Lorsque vous concevez votre stratégie d'autorisations, vous devez créer des politiques avec des instructions `Allow` (Autoriser) pour permettre à vos principaux d'effectuer des demandes avec succès. Toutefois, vous pouvez choisir n'importe quelle combinaison de refus explicites et implicites.

Par exemple, vous pouvez créer la politique suivante qui inclut des actions autorisées, des actions implicitement rejetées et des actions explicitement rejetées. L'instruction `AllowGetList` permet un accès en lecture seule aux actions IAM qui commencent par les préfixes `Get` et `List`. Toutes les autres actions dans IAM, comme `iam:CreatePolicy`, sont implicitement rejetées. L'instruction `DenyReports` rejette explicitement l'accès aux rapports IAM en rejetant l'accès aux actions qui incluent le suffixe `Report`, comme `iam:GetOrganizationsAccessReport`. Si quelqu'un ajoute une autre politique à ce principal pour lui donner accès aux rapports IAM, comme `iam:GenerateCredentialReport`, les demandes liées aux rapports sont toujours rejetées en raison de ce rejet explicite.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowGetList",
            "Effect": "Allow",
            "Action": [
                "iam:Get*",
                "iam:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "DenyReports",
            "Effect": "Deny",
```

```
            "Action": "iam:*Report",
            "Resource": "*"
        }
    ]
}
```

Logique d'évaluation des politiques entre comptes

Vous pouvez autoriser un principal d'un compte à accéder aux ressources d'un second compte. C'est ce que l'on appelle l'accès entre comptes. Lorsque vous autorisez un accès entre comptes, le compte dans lequel se trouve le principal est appelé compte approuvé. Le compte dans lequel se trouve la ressource est le compte d'approbation.

Pour autoriser l'accès entre comptes, vous associez une politique basée sur les ressources à la ressource que vous souhaitez partager. Vous devez également attacher une politique basée sur l'identité à l'identité qui fait office de principal dans la demande. La politique basée sur les ressources dans le compte d'approbation doit spécifier le principal du compte approuvé qui aura accès à la ressource. Vous pouvez spécifier l'intégralité du compte ou ses utilisateurs IAM, les utilisateurs fédérés, les rôles IAM ou les sessions à rôles endossés. Vous pouvez également spécifier un AWS service en tant que principal. Pour plus d'informations, consultez Spécification d'un principal.

La politique basée sur l'identité du principal doit permettre l'accès demandé à la ressource dans le service d'approbation. Vous pouvez aboutir à cela en spécifiant l'ARN de la ressource ou en autorisant l'accès à toutes les ressources (*).

Dans IAM, vous pouvez associer une politique basée sur les ressources à un rôle IAM pour permettre aux principaux d'autres comptes d'endosser ce rôle. La politique basée sur les ressources du rôle est appelée politique d'approbation de rôle. Une fois qu'ils endossent ce rôle, les principaux autorisés peuvent utiliser les informations d'identification temporaires résultantes pour accéder à plusieurs ressources de votre compte. Cet accès est défini dans la politique d'autorisations basée sur l'identité du rôle. Pour connaître les différences entre l'autorisation d'accès entre comptes à l'aide de rôles et l'autorisation d'accès entre comptes à l'aide d'autres politiques basées sur les ressources, veuillez consulter Accès intercompte aux ressources dans IAM.

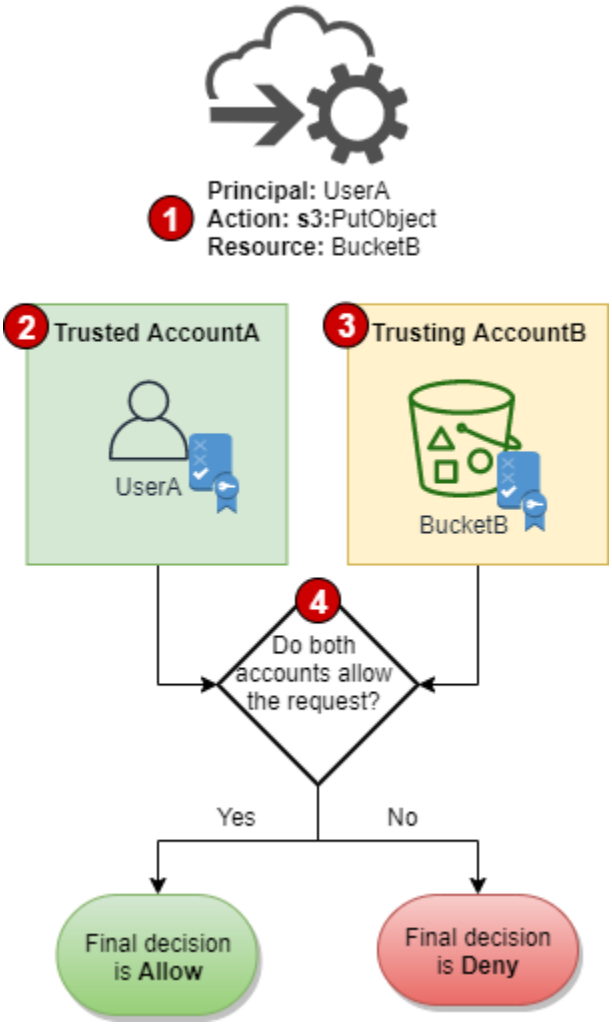
> **Important**
>
> D'autres services peuvent influer sur la logique d'évaluation des politiques. Par exemple, AWS Organizations prend en charge les politiques de contrôle des services qui peuvent être appliquées à un ou plusieurs comptes principaux. AWS Resource Access Manager prend en

charge les fragments de politique qui contrôlent les actions que les principaux sont autorisés à effectuer sur les ressources partagées avec eux.

Comment déterminer si une demande d'accès entre comptes est autorisée

Pour les demandes entre comptes, le demandeur du compte approuvé `AccountA` doit avoir une politique basée sur l'identité. Cette politique doit lui permettre de faire une demande à la ressource du compte d'approbation `AccountB`. En outre, la politique basée sur les ressources du compte `AccountB` doit autoriser le demandeur du compte `AccountA` à accéder à la ressource.

Lorsque vous faites une demande entre comptes, AWS effectue deux évaluations. AWS évalue la demande dans le compte de confiance et le compte de confiance. Pour de plus amples informations sur la façon dont une demande est évaluée au sein d'un seul compte, veuillez consulter Identification d'une demande autorisée ou refusée dans un compte. La demande n'est autorisée que si les deux évaluations renvoient une décision `Allow`.



1. Lorsqu'un principal d'un compte fait une demande d'accès à une ressource d'un autre compte, il s'agit d'une demande entre comptes.
2. Le principal demandeur se trouve dans le compte approuvé (`AccountA`). Lorsque AWS évalue ce compte, il vérifie la politique basée sur l'identité et toutes les politiques pouvant limiter une politique basée sur l'identité. Pour plus d'informations, veuillez consulter Évaluation des politiques dans un compte unique.
3. La ressource demandée se trouve dans le compte d'approbation (`AccountB`). Lorsque AWS évalue ce compte, il vérifie la politique basée sur les ressources qui est associée à la ressource demandée et toutes les politiques pouvant limiter une politique basée sur les ressources. Pour plus d'informations, consultez Évaluation des politiques dans un compte unique.
4. AWS autorise la demande uniquement si les deux évaluations de la politique du compte l'autorisent.

Exemple d'évaluation de politique entre comptes

L'exemple suivant illustre un scénario dans lequel un utilisateur d'un compte se voit accorder des autorisations par une politique basée sur les ressources dans un second compte.

Supposons que Carlos est un développeur dont le nom d'utilisateur IAM est `carlossalazar` dans le compte 111111111111. Il veut enregistrer un fichier dans le compartiment Amazon S3 `Production-logs` du compte 222222222222.

Supposons également que la politique suivante soit attachée à l'utilisateur IAM `carlossalazar`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        },
        {
            "Sid": "AllowS3ProductionObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object*",
            "Resource": "arn:aws:s3:::Production/*"
        },
```

```
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::*log*",
                "arn:aws:s3:::*log*/*"
            ]
        }
    ]
}
```

L'instruction `AllowS3ListRead` de cette politique autorise Carlos à afficher une liste de tous les compartiments dans Amazon S3. L'instruction `AllowS3ProductionObjectActions` offre à Carlos un accès complet aux objets dans le compartiment `Production`. L'instruction `DenyS3Logs` refuse à Carlos l'accès à n'importe quel compartiment S3 comprenant `log` dans son nom. Elle refuse également l'accès à tous les objets de ces compartiments.

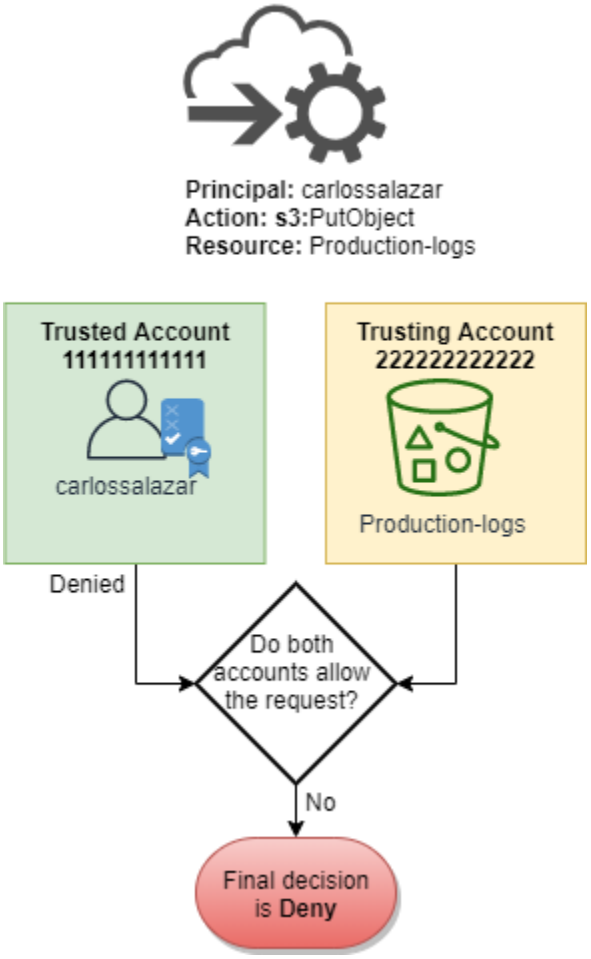
De plus, la politique basée sur les ressources suivante (appelée politique de compartiment) est attachée au compartiment `Production` dans le compte 222222222222.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:PutObject*",
                "s3:ReplicateObject",
                "s3:RestoreObject"
            ],
            "Principal": { "AWS": "arn:aws:iam::111111111111:user/carlossalazar" },
            "Resource": "arn:aws:s3:::Production/*"
        }
    ]
}
```

Cette politique permet à l'utilisateur `carlossalazar` d'accéder aux objets du compartiment `Production`. Il peut créer et éditer, mais pas supprimer les objets dans le compartiment. Il ne peut pas gérer le compartiment lui-même.

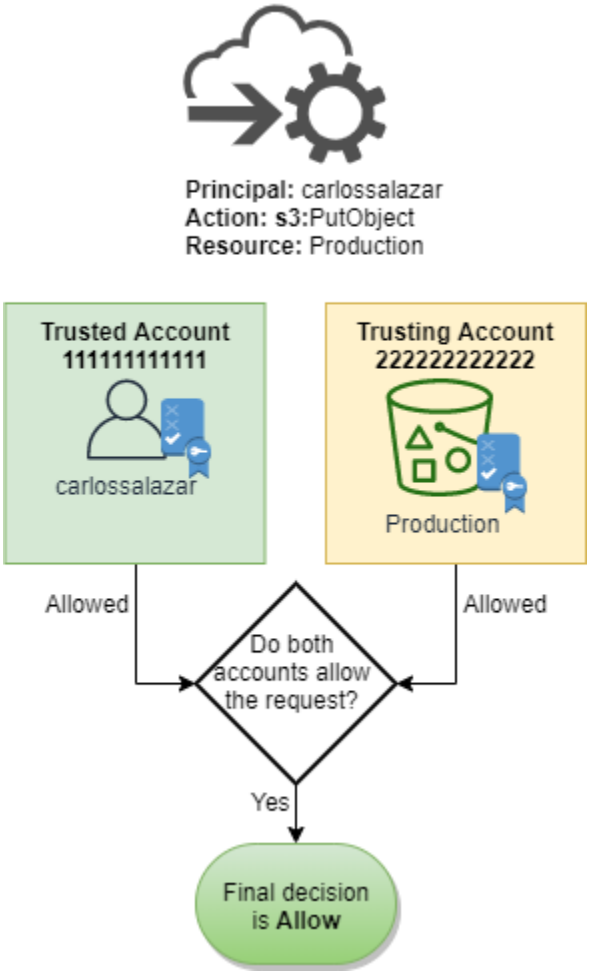
Lorsque Carlos effectue une demande d'enregistrement d'un fichier dans le compartiment `Production-logs`, AWS détermine les stratégies qui s'appliquent à la demande. Dans ce cas, la politique basée sur l'identité associée à l'utilisateur `carlossalazar` est la seule politique qui s'applique dans le compte `111111111111`. Dans le compte `222222222222`, il n'y a pas de politique basée sur les ressources associée au compartiment `Production-logs`. Lorsque AWS évalue le compte `111111111111`, il renvoie une décision `Deny`. En effet, l'instruction `DenyS3Logs` de la politique basée sur l'identité refuse explicitement l'accès à tous les compartiments de journaux. Pour de plus amples informations sur la façon dont une demande est évaluée au sein d'un seul compte, veuillez consulter Identification d'une demande autorisée ou refusée dans un compte.

Étant donné que la demande est explicitement refusée dans l'un des comptes, la décision finale est de refuser la demande.



Supposons que Carlos réalise alors son erreur et essaie d'enregistrer le fichier dans le `Production` bucket. AWS vérifie d'abord `111111111111` le compte pour déterminer si la demande est autorisée. Seule la politique basée sur l'identité s'applique et autorise la demande. AWS puis vérifie le compte`222222222222`. Seule la politique basée sur les ressources associée au compartiment

`Production` s'applique et elle autorise la demande. Étant donné que les deux comptes autorisent la demande, la décision finale est d'autoriser la demande.



Syntaxe du langage de politique JSON IAM

Cette page présente la syntaxe formelle du langage de stratégie utilisé pour créer des politiques JSON dans IAM. Nous vous la présentons pour vous permettre de mieux comprendre comment créer et valider des politiques.

Pour consulter des exemples de politiques, reportez-vous aux rubriques suivantes :

- Politiques et autorisations dans IAM
- Exemples de politiques basées sur l'identité IAM
- Exemples de politiques relatives à l'utilisation de la console Amazon EC2 et exemples de politiques relatives à l'utilisation de la AWS CLI, de l'interface de ligne de commande Amazon EC2 ou AWS d'un SDK dans le guide de l'utilisateur Amazon EC2.

- Exemples de politique de compartiment et Exemples de politiques utilisateur dans le guide de l'utilisateur service de stockage simple Amazon.

Pour obtenir des exemples de politiques utilisées dans d'autres AWS services, consultez la documentation de ces services.

Rubriques

- Langage de politique et JSON
- Conventions utilisées dans cette syntaxe
- Syntaxe
- Remarques sur la syntaxe de politique

Langage de politique et JSON

Les politiques sont créées à l'aide du format JSON. Lorsque vous créez ou modifiez une politique JSON, IAM peut effectuer une validation de politique pour vous aider à créer une politique efficace. IAM identifie les erreurs de syntaxe JSON, tandis que IAM Access Analyzer fournit des vérifications de politique supplémentaires avec des recommandations pour vous aider à affiner vos politiques. Pour en savoir plus sur la validation de politiques, veuillez consulter Validation de politiques IAM. Pour en savoir plus sur les vérifications des politiques IAM Access Analyzer et les recommandations exploitables, veuillez consulter Validation de politique IAM Access Analyzer.

Ce document n'entend pas fournir une description complète de ce qui constitue un code JSON valide. Toutefois, voici quelques règles de base applicables à JSON :

- Les espaces entre des entités individuelles sont autorisés.
- Les valeurs sont entourées de guillemets. Les guillemets sont facultatifs pour les valeurs numériques et booléennes.
- De nombreux éléments (par exemple, `action_string_list` et `resource_string_list`) acceptent une valeur composée d'un tableau JSON. Les tableaux peuvent inclure une ou plusieurs valeurs. Lorsqu'un tableau comporte plusieurs valeurs, il est entouré de crochets (`[` et `]`) et les valeurs sont séparées par une virgule, comme dans l'exemple suivant :

  ```
  "Action" : ["ec2:Describe*","ec2:List*"]
  ```

- Les types de données JSON de base (booléen, nombre et chaîne) sont définis dans RFC 7159.

Conventions utilisées dans cette syntaxe

Les conventions suivantes sont utilisées dans cette syntaxe :

- Les caractères suivants sont des jetons JSON et sont inclus dans les politiques :

  ```
  { } [ ] " , :
  ```
- Les caractères suivants sont des caractères spéciaux dans la syntaxe et ne sont pas inclus dans les politiques :

  ```
  = < > ( ) |
  ```
- Si un élément autorise plusieurs valeurs, ceci est indiqué à l'aide de valeurs répétées, d'une virgule de délimitation et de points de suspension (`...`). Exemples :

  ```
  [<action_string>, <action_string>, ...]
  ```

  ```
  <principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }
  ```

  Si plusieurs valeurs sont autorisées, il est également possible d'en inclure une seule. Lors de l'utilisation d'une seule valeur, la virgule de fin doit être omise. Si l'élément accepte un tableau (entouré de [et]) mais que vous n'utilisez qu'une valeur, les crochets sont facultatifs. Exemples :

  ```
  "Action": [<action_string>]
  ```

  ```
  "Action": <action_string>
  ```
- Un point d'interrogation (?) placé après un élément indique que cet élément est facultatif. Exemple :

  ```
  <version_block?>
  ```

  Toutefois, reportez-vous aux notes relatives à la syntaxe ci-après pour plus de détails sur les éléments facultatifs.
- Une ligne verticale (|) entre des éléments indique plusieurs choix. Dans la syntaxe, les parenthèses définissent la portée de ces choix. Exemple :

  ```
  ("Principal" | "NotPrincipal")
  ```
- Les éléments qui doivent être des chaînes littérales sont entourés de guillemets doubles ("). Exemple :

  ```
  <version_block> = "Version" : ("2008-10-17" | "2012-10-17")
  ```

Pour des informations complémentaires, reportez-vous à la section Remarques sur la syntaxe de politique après la description de la syntaxe.

Syntaxe

La liste suivante décrit la syntaxe du langage de politique. Pour plus d'informations sur les conventions utilisées dans la liste, reportez-vous à la section précédente. Pour des informations complémentaires, reportez-vous aux notes ci-après.

> **Note**
>
> Cette syntaxe décrit des politiques marquées avec une version `2008-10-17` et `2012-10-17`. Un élément de politique `Version` varie d'une version de politique. L'élément de politique `Version` est utilisé dans une politique pour définir la version de la langue de la politique. En revanche, une version de politique est créée lorsque vous apportez des modifications à une politique gérée par le client dans IAM. La politique modifiée ne remplace pas la politique existante. À la place, IAM crée une nouvelle version de la politique gérée. Pour en savoir plus sur l'élément de politique `Version`, consultez Éléments de politique JSON IAM : Version. Pour en savoir plus sur les versions de politiques, consultez the section called “Gestion des versions des politiques IAM”.

```
policy  = {
     <version_block?>
     <id_block?>
     <statement_block>
}

<version_block> = "Version" : ("2008-10-17" | "2012-10-17")

<id_block> = "Id" : <policy_id_string>

<statement_block> = "Statement" : [ <statement>, <statement>, ... ]

<statement> = {
    <sid_block?>,
    <principal_block?>,
    <effect_block>,
    <action_block>,
    <resource_block>,
```

```
    <condition_block?>
}

<sid_block> = "Sid" : <sid_string>

<effect_block> = "Effect" : ("Allow" | "Deny")

<principal_block> = ("Principal" | "NotPrincipal") : ("*" | <principal_map>)

<principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }

<principal_map_entry> = ("AWS" | "Federated" | "Service" | "CanonicalUser") :
    [<principal_id_string>, <principal_id_string>, ...]

<action_block> = ("Action" | "NotAction") :
    ("*" | [<action_string>, <action_string>, ...])

<resource_block> = ("Resource" | "NotResource") :
    : ("*" | <resource_string> | [<resource_string>, <resource_string>, ...])

<condition_block> = "Condition" : { <condition_map> }
<condition_map> = {
  <condition_type_string> : { <condition_key_string> : <condition_value_list> },
  <condition_type_string> : { <condition_key_string> : <condition_value_list> }, ...
}
<condition_value_list> = [<condition_value>, <condition_value>, ...]
<condition_value> = (<condition_value_string> | <condition_value_string> |
 <condition_value_string>)
```

Remarques sur la syntaxe de politique

- Une politique peut contenir un tableau de instructions.
- La taille maximale des politiques est comprise entre 2 048 et 10 240 caractères, selon l'entité à laquelle la politique est attachée. Pour plus d'informations, veuillez consulter IAM et quotas AWS STS. Lors du calcul de la taille de la politique, les espaces ne sont pas inclus.
- Des éléments individuels ne doivent pas contenir plusieurs instances de la même clé. Par exemple, vous ne pouvez pas inclure le bloc `Effect` à deux reprises dans la même instruction.
- Les blocs peuvent être insérés dans n'importe quel ordre. Par exemple, `version_block` peut suivre `id_block` dans une politique. De la même façon `effect_block`, `principal_block` et `action_block` peuvent figurer dans n'importe quel ordre dans une instruction.

- L'élément `id_block` est facultatif dans les politiques basées sur les ressources. Il ne doit pas être inclus dans les stratégies basées sur une identité.
- L'élément `principal_block` est obligatoire dans les politiques basées sur les ressources (par exemple, dans les politiques de compartiment Amazon S3) et dans les politiques d'approbation de rôles IAM. Il ne doit pas être inclus dans les stratégies basées sur une identité.
- L'élément `principal_map` des politiques de compartiment Amazon S3 peut inclure l'ID `CanonicalUser`. La plupart des politiques basées sur les ressources ne prennent pas en charge ce mappage. Pour en savoir plus sur l'utilisation de l'ID d'utilisateur canonique dans une politique de compartiment, veuillez consulter spécification d'un principal dans une politique dans le guide de l'utilisateur service de stockage simple Amazon.
- Chaque valeur de chaîne (`policy_id_string`, `sid_string`, `principal_id_string`, `action_string`, `resource_string`, `condition_type_string`, `condition_key_string` et la version de chaîne de `condition_value`) peut avoir ses propres limites de longueur minimale et maximale, des valeurs autorisées spécifiques ou un format interne obligatoire.

Remarques à propos de valeurs de chaîne

Cette section contient des informations complémentaires sur les valeurs de chaîne utilisées dans les différents éléments d'une politique.

`action_string`

Comporte un espace de noms de service, deux points et le nom d'une action. Un nom d'action peut inclure des caractères génériques. Exemples :

```
"Action":"ec2:StartInstances"

"Action":[
  "ec2:StartInstances",
  "ec2:StopInstances"
]

"Action":"cloudformation:*"

"Action":"*"

"Action":[
  "s3:Get*",
  "s3:List*"
```

```
]
```

policy_id_string

Permet d'inclure des informations globales sur la politique. Certains services tels qu'Amazon SQS et Amazon SNS utilisent l'élément `Id` en tant qu'élément réservé. Si aucune restriction n'est imposée par un service individuel, policy_id_string peut inclure des espaces. Certains services exigent que cette valeur soit unique dans un compte AWS .

> **Note**
>
> L'élément `id_block` est autorisé dans les politiques basées sur les ressources, mais pas dans celles basées sur une identité.

La longueur n'est pas limitée, mais cette chaîne contribue à la longueur totale de la politique, qui elle est limitée.

```
"Id":"Admin_Policy"

"Id":"cd3ad3d9-2776-4ef1-a904-4c229d1642ee"
```

sid_string

Permet d'inclure des informations relatives à une instruction individuelle. Pour les politiques IAM, les caractères alphanumériques de base (A-Z, a-z, 0-9) sont les seuls autorisés dans la valeur `Sid`. Les autres services AWS prenant en charge les politiques de ressources peuvent imposer d'autres exigences pour la valeur `Sid`. Par exemple, certains services exigent que cette valeur soit unique dans un Compte AWS, et certains services autorisent des caractères supplémentaires tels que des espaces dans la `Sid` valeur.

```
"Sid":"1"

"Sid": "ThisStatementProvidesPermissionsForConsoleAccess"
```

principal_id_string

Permet de spécifier un principal à l'aide du nom de ressource Amazon (ARN) de l'utilisateur IAM Compte AWS, du rôle IAM, de l'utilisateur fédéré ou de l'utilisateur assumant le rôle. Pour an

Compte AWS, vous pouvez également utiliser la forme abrégée `AWS:accountnumber` au lieu de l'ARN complet. Pour toutes les options, notamment les services AWS ;, les rôles endossés et ainsi de suite, consultez Spécification d'un principal.

Notez que vous pouvez utiliser * uniquement pour spécifier « tout le monde/anonyme ». Vous ne pouvez pas l'utiliser pour spécifier une partie de nom ou d'ARN.

`resource_string`

Dans la plupart des cas, la valeur est composée d'un Amazon Resource Name (ARN).

```
"Resource":"arn:aws:iam::123456789012:user/Bob"

"Resource":"arn:aws:s3:::examplebucket/*"
```

`condition_type_string`

Identifie le type de condition testé, par exemple `StringEquals`, `StringLike`, `NumericLessThan`, `DateGreaterThanEquals`, `Bool`, `BinaryEquals`, `IpAddress`, `ArnEquals`, etc. Pour obtenir la liste complète des types de conditions, consultez Éléments de politique JSON IAM : Opérateurs de condition.

```
"Condition": {
  "NumericLessThanEquals": {
    "s3:max-keys": "10"
  }
}

"Condition": {
  "Bool": {
    "aws:SecureTransport": "true"
  }
}

"Condition": {
  "StringEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
}
```

`condition_key_string`

Identifie la clé de condition dont la valeur sera testée pour déterminer si la condition est remplie. AWS définit un ensemble de clés de condition disponibles dans tous les AWS services, y compris `aws:PrincipalTypeaws:SecureTransport`, et`aws:userid`.

Pour obtenir la liste des clés de AWS condition, voirAWS clés contextuelles de condition globale. Pour connaître les clés de condition spécifiques à un service, reportez-vous à la documentation relative au service, notamment :

- Spécification de conditions dans une politique dans le guide de l'utilisateur service de stockage simple Amazon
- Politiques IAM pour Amazon EC2 dans le guide de l'utilisateur Amazon EC2.

```
"Condition":{
  "Bool": {
      "aws:SecureTransport": "true"
   }
}

"Condition": {
  "StringNotEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
}

"Condition": {
  "StringEquals": {
    "aws:ResourceTag/purpose": "test"
  }
}
```

`condition_value_string`

Identifie la valeur de la chaîne condition_key_string qui détermine si la condition est remplie. Pour obtenir la liste complète des valeurs valides pour un type de condition, consultez Éléments de politique JSON IAM : Opérateurs de condition.

```
"Condition":{
  "ForAnyValue:StringEquals": {
  "dynamodb:Attributes": [
   "ID",
```

```
    "PostDateTime"
          ]
  }
}
```

AWS politiques gérées pour les fonctions professionnelles

Nous vous recommandons d'utiliser des politiques qui accordent le moins de privilèges ou d'accorder uniquement les autorisations requises pour effectuer une tâche. Le moyen le plus sûr d'octroyer le moindre privilège consiste à écrire une politique personnalisée contenant uniquement les autorisations requises par votre équipe. Vous devez créer un processus pour autoriser votre équipe à demander plus d'autorisations si nécessaire. Il faut du temps et de l'expertise pour créer des politiques gérées par le client IAM qui ne fournissent à votre équipe que les autorisations dont elle a besoin.

Pour commencer à ajouter des autorisations à vos identités IAM (utilisateurs, groupes d'utilisateurs et rôles), vous pouvez utiliserAWS politiques gérées. AWS les politiques gérées couvrent les cas d'utilisation courants et sont disponibles dans votre Compte AWS. AWS les politiques gérées n'accordent pas les autorisations du moindre privilège. Vous devez prendre en compte le risque de sécurité constitué par l'octroi, à vos principaux, de davantage d'autorisations que nécessaire pour accomplir leur tâche.

Vous pouvez associer des politiques AWS gérées, y compris des fonctions de travail, à n'importe quelle identité IAM. Pour passer aux autorisations du moindre privilège, vous pouvez exécuter AWS Identity and Access Management Access Analyzer pour surveiller les principaux à l'aide de politiques AWS gérées. Lorsque vous savez quelles autorisations ils utilisent, vous pouvez écrire une politique personnalisée ou générer une politique avec uniquement les autorisations requises pour votre équipe. Cela est moins sûr, mais offre plus de flexibilité à mesure que vous apprenez comment votre équipe les utilise AWS.

AWS les politiques gérées pour les fonctions professionnelles sont conçues pour s'aligner étroitement sur les fonctions professionnelles courantes dans le secteur informatique. Vous pouvez utiliser ces politiques pour accorder les autorisations nécessaires afin d'exécuter les tâches prévues de la part quelqu'un occupant une fonction spécifique. Ces politiques regroupent les autorisations pour de nombreux services dans une seule politique plus facile à utiliser que des autorisations dispersées dans de nombreuses politiques.

Utiliser des rôles pour combiner les services

Certaines politiques utilisent des rôles de service IAM pour vous aider à tirer parti des fonctionnalités d'autres AWS services. Ces politiques accordent l'accès à un service`iam:passrole`, ce qui permet à l'utilisateur disposant de cette politique de transmettre un rôle à un AWS service. Ce rôle délègue les autorisations IAM au AWS service pour effectuer des actions en votre nom.

Vous devez créer les rôles selon vos besoins. Par exemple, la politique d'administrateur réseau permet à un utilisateur disposant de cette politique de transmettre un rôle nommé « flow-logs-vpc » au service Amazon. CloudWatch CloudWatch utilise ce rôle pour enregistrer et capturer le trafic IP pour les VPC créés par l'utilisateur.

Pour s'accorder aux meilleures pratiques de sécurité, les politiques pour les activités professionnelles incluent des filtres qui limitent les noms de rôles valides possibles à transmettre. Cela permet d'éviter d'accorder des autorisations inutiles. Si vos utilisateurs ont besoin des rôles de services facultatifs, vous devez créer un rôle qui suit la convention d'affectation de noms spécifiée dans la politique. Vous pouvez ensuite accorder des autorisations pour le rôle. L'utilisateur peut alors configurer le service pour utiliser ce rôle, et lui octroyer toutes les autorisations fournies par le rôle.

Dans les sections suivantes, chaque nom de politique comporte un lien vers la page des détails de la politique dans AWS Management Console. Vous pouvez alors consulter le document de politique et examiner les autorisations qu'il accorde.

Fonction de tâche Administrateur

AWS nom de la politique gérée : AdministratorAccess

Cas d'utilisation : cet utilisateur a un accès total et peut déléguer des autorisations à tous les services et toutes les ressources dans AWS.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour des politiques AWS gérées pour les fonctions professionnelles.

Description de la politique : Cette politique autorise toutes les actions pour tous les AWS services et pour toutes les ressources du compte. Pour plus d'informations sur la stratégie gérée, consultez le Guide AdministratorAccessde référence des politiques AWS gérées.

> **Note**
>
> Avant qu'un utilisateur ou un rôle IAM puisse accéder à la AWS Billing and Cost Management console avec les autorisations définies dans cette politique, vous devez d'abord activer l'accès aux utilisateurs et aux rôles IAM. Pour ce faire, suivez les instructions de l'étape 1 du didacticiel portant sur la délégation de l'accès à la console de facturation.

Fonction de tâche Facturation

AWS nom de la politique gérée : Facturation

Cas d'utilisation : cet utilisateur a besoin d'afficher les informations de facturation, de préparer les paiements et d'autoriser les paiements. L'utilisateur peut surveiller les coûts cumulés pour l'ensemble du AWS service.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour des politiques AWS gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde la totalité des autorisations de gestion de la facturation, des coûts, des moyens de paiement et des rapports. Pour des exemples de politiques de gestion des coûts supplémentaires, consultez les exemples AWS Billing de politiques dans le Guide de AWS Billing and Cost Management l'utilisateur. Pour plus d'informations sur la politique gérée, consultez le Guide de référence sur la facturation dans les politiques AWS gérées.

> **Note**
>
> Avant qu'un utilisateur ou un rôle IAM puisse accéder à la AWS Billing and Cost Management console avec les autorisations définies dans cette politique, vous devez d'abord activer l'accès aux utilisateurs et aux rôles IAM. Pour ce faire, suivez les instructions de l'étape 1 du didacticiel portant sur la délégation de l'accès à la console de facturation.

Fonction de tâche Administrateur de base de données

AWS nom de la politique gérée : DatabaseAdministrator

Cas d'utilisation : cet utilisateur installe, configure et gère des bases de données dans le AWS cloud.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour des politiques AWS gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde les autorisations de créer, configurer et gérer des bases de données. Il inclut l'accès aux services AWS de base de données, tels qu'Amazon DynamoDB, Amazon Relational Database Service (RDS) et Amazon Redshift. Consultez la politique pour la liste entière de services de base de données que prend en charge cette politique. Pour plus d'informations sur la stratégie gérée, consultez le Guide DatabaseAdministratorde référence des politiques AWS gérées.

Cette politique relative aux fonctions professionnelles favorise la capacité de transférer des rôles aux AWS services. Elle autorise l'action `iam:PassRole` seulement pour les rôles nommés dans le tableau suivant. Pour plus d'informations, consultez Création des rôles et association des politiques (console) plus loin dans cette rubrique.

Rôles de services IAM facultatifs pour la fonction d'administrateur de base de données

Cas d’utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	Sélectionnez cette politique AWS gérée
Permettre à l'utilisateur de surveiller des bases de données RDS	rds-monitoring-role	Rôle Amazon RDS pour la surveillance améliorée	Rôle Amazon RDS EnhancedMonitoring
Permettre AWS Lambda de surveiller votre base de données et d'accéder à des bases de données externes	rdbms-lambda-access	Amazon EC2	AWSLambda_FullAccess
Autoriser Lambda à télécharger des fichiers vers Amazon S3 et vers	lambda_exec_role	AWS Lambda	Créer une nouvelle politique gérée, tel que

Cas d’utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	Sélectionnez cette politique AWS gérée
des clusters Amazon Redshift avec DynamoDB			défini dans l'AWS Big Data Blog
Autoriser les fonctions Lambda à agir comme des déclencheurs pour vos tables DynamoDB	lambda-dynamodb-*	AWS Lambda	AWSLambda DynamoDBExecutionRole
Autoriser les fonctions Lambda à accéder à Amazon RDS dans un VPC	lambda-vpc-execution-role	Créer un rôle avec une politique d'approbation, tel que défini dans le Guide du développeur AWS Lambda	AWSLambda VPCAccessExecutionRole
AWS Data Pipeline Autoriser l'accès à vos AWS ressources	DataPipelineDefaultRole	Créer un rôle avec une politique d'approbation, tel que défini dans le Guide du développeur AWS Data Pipeline	La AWS Data Pipeline documentation répertorie les autorisations requises pour ce cas d'utilisation. Voir les rôles IAM pour AWS Data Pipeline

Cas d'utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	Sélectionnez cette politique AWS gérée
Permettre à vos applications qui s'exécutent sur des instances Amazon EC2 d'accéder à vos ressources AWS	DataPipelineDefaultResourceRôle	Créer un rôle avec une politique d'approbation, tel que défini dans le Guide du développeur AWS Data Pipeline	Amazon EC2 RoleforData PipelineRole

Fonction de tâche Scientifique des données

AWS nom de la politique gérée : DataScientist

Cas d'utilisation : cet utilisateur exécute des tâches et des demandes Hadoop. L'utilisateur accède également à des informations pour l'analytique des données et la business intelligence, et analyse celles-ci.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour des politiques AWS gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde les autorisations nécessaires pour créer, gérer et exécuter des requêtes sur un cluster Amazon EMR et pour effectuer des analyses de données à l'aide d'outils tels qu'Amazon. QuickSight La politique inclut l'accès à des services de data scientist supplémentaires AWS Data Pipeline, tels qu'Amazon EC2, Amazon Kinesis, Amazon Machine Learning et. SageMaker Consultez la politique pour la liste entière de services de scientifique de données que prend en charge cette politique. Pour plus d'informations sur la stratégie gérée, consultez le Guide DataScientistde référence des politiques AWS gérées.

Cette politique relative aux fonctions professionnelles favorise la capacité de transférer des rôles aux AWS services. Une instruction permet de transmettre n'importe quel rôle à SageMaker. Une

autre instruction autorise l'action `iam:PassRole` seulement pour les rôles nommés dans le tableau suivant. Pour plus d'informations, consultez Création des rôles et association des politiques (console) plus loin dans cette rubrique.

Rôles de services IAM facultatifs pour la fonction de spécialiste des données

Cas d'utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	AWS politique gérée à sélectionner
Autoriser les instances Amazon EC2 à accéder aux services et aux ressources appropriés pour les clusters	EMR-EC2_ DefaultRole	Amazon EMR pour EC2	AmazonElasticMapReduceforRôle EC2
Autoriser Amazon EMR à accéder au service et aux ressources Amazon EC2 pour les clusters	EMR_ DefaultRole	Amazon EMR	Amazon EMR _v2 ServicePolicy
Autoriser le service géré Kinesis pour Apache Flink à accéder aux sources de données de diffusion	kinesis-*	Créer un rôle avec une politique d'approbation, tel que défini dans l'AWS Big Data Blog.	Consultez l'AWS Big Data Blog, qui définit les quatre options possibles en fonction de votre cas d'utilisation.
AWS Data Pipeline Autoriser l'accès à vos AWS ressources	DataPipelineDefaultRole	Créer un rôle avec une politique d'approbation, tel que défini dans le Guide du développeur AWS Data Pipeline	La AWS Data Pipeline documentation répertorie les autorisations requises pour ce cas d'utilisation. Voir les rôles IAM pour AWS Data Pipeline

Cas d’utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	AWS politique gérée à sélectionner
Permettre à vos applications qui s'exécutent sur des instances Amazon EC2 d'accéder à vos ressources AWS	DataPipelineDefaultResourceRôle	Créer un rôle avec une politique d'approbation, tel que défini dans le Guide du développeur AWS Data Pipeline	Amazon EC2 RoleforData PipelineRole

Fonction de tâche Utilisateur avec pouvoir Développeur

AWS nom de la politique gérée : PowerUser Access

Cas d'utilisation : cet utilisateur exécute des tâches de développement d'applications et peut créer et configurer des ressources et des services qui prennent en charge le développement d'applications AWS conscientes.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour des politiques AWS gérées pour les fonctions professionnelles.

Description de la politique : La première déclaration de cette politique utilise l'`NotAction`élément pour autoriser toutes les actions pour tous les AWS services et pour toutes les ressources AWS Identity and Access Management, à l'exception de AWS Organizations, et AWS Account Management. La seconde instruction accorde des autorisations IAM pour créer un rôle lié à un service. Elle est requise par certains services qui doivent accéder aux ressources d'un autre service, par exemple, un compartiment Amazon S3. Elle accorde également autorisations Organizations pour afficher des informations sur l'organisation de l'utilisateur, en particulier l'adresse e-mail du compte de gestion et les limitations de l'organisation. Bien que cette stratégie limite l'accès à IAM, Organizations, elle permet à l'utilisateur d'effectuer toutes les actions si IAM Identity Center est activé. Il accorde

également à la gestion du compte des autorisations permettant de voir quelles AWS régions sont activées ou désactivées pour le compte.

Fonction de tâche Administrateur réseau

AWS nom de la politique gérée : NetworkAdministrator

Cas d'utilisation : cet utilisateur est chargé de configurer et de gérer les ressources AWS du réseau.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour des politiques AWS gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde des autorisations pour créer et gérer des ressources réseau dans Auto Scaling, Amazon EC2 AWS Direct Connect, Route 53, Amazon CloudFront, Elastic Load Balancing AWS Elastic Beanstalk, Amazon SNS CloudWatch, Logs CloudWatch , Amazon S3, IAM et Amazon Virtual Private Cloud. Pour plus d'informations sur la stratégie gérée, consultez le Guide NetworkAdministratorde référence des politiques AWS gérées.

Cette fonction nécessite la capacité de transmettre des rôles aux AWS services. Elle accorde `iam:GetRole` et `iam:PassRole` seulement pour les rôles nommés dans le tableau suivant. Pour plus d'informations, consultez Création des rôles et association des politiques (console) plus loin dans cette rubrique.

Rôles de services IAM facultatifs pour la fonction d'administrateur réseau

Cas d’utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	AWS politique gérée à sélectionner
Permet à Amazon VPC de créer et de gérer les CloudWatch journaux au nom de l'utilisateur afin de surveiller le trafic IP entrant et sortant de votre VPC	flow-logs-*	Créer un rôle avec une politique d'approbation, tel que défini dans le Guide de l'utilisateur Amazon VPC	Ce cas d'utilisation n'a pas de politique AWS gérée existante, mais la documentation répertorie les autorisations

Cas d’utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	AWS politique gérée à sélectionner
			requises. Consultez le Guide de l'utilisateur Amazon VPC.

Accès en lecture seule

AWS nom de la politique gérée : ReadOnly Access

Cas d'utilisation : cet utilisateur nécessite un accès en lecture seule à toutes les ressources d'un Compte AWS.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour des politiques AWS gérées pour les fonctions professionnelles.

Description de la politique : cette politique octroie les autorisations de répertorier, obtenir, décrire ou afficher les ressources et leurs attributs. Elle n'inclut pas des fonctions de mutation, telles que créer ou supprimer. Cette politique inclut l'accès en lecture seule aux AWS services liés à la sécurité, tels que et. AWS Identity and Access Management AWS Billing and Cost Management Consultez la politique pour la liste entière des services et actions que prend en charge cette politique.

Fonction de tâche Audit de sécurité

AWS nom de la politique gérée : SecurityAudit

Cas d'utilisation : cet utilisateur surveille les comptes pour vérifier leur conformité aux exigences de sécurité. Il peut accéder aux journaux et aux événements pour rechercher des brèches de sécurité potentielles ou d'éventuelles activités malveillantes.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les

mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour des politiques AWS gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde des autorisations pour consulter les données de configuration de nombreux AWS services et pour consulter leurs journaux. Pour plus d'informations sur la stratégie gérée, consultez le Guide SecurityAuditde référence des politiques AWS gérées.

Fonction de tâche Utilisateur support

AWS nom de la politique gérée : SupportUser

Cas d'utilisation : cet utilisateur contacte le AWS Support, crée des dossiers d'assistance et consulte l'état des dossiers existants.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour des politiques AWS gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde des autorisations pour créer et mettre à jour AWS Support des dossiers. Pour plus d'informations sur la stratégie gérée, consultez le Guide SupportUserde référence des politiques AWS gérées.

Fonction de tâche Administrateur système

AWS nom de la politique gérée : SystemAdministrator

Cas d'utilisation : cet utilisateur configure et gère les ressources pour les opérations de développement.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour des politiques AWS gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde des autorisations pour créer et gérer des ressources sur une grande variété de AWS services AWS CloudTrail, notamment Amazon CloudWatch, AWS CodeCommit, AWS CodeDeploy, AWS Config, AWS Directory Service,, Amazon EC2,, AWS Identity and Access Management, AWS Key Management Service, AWS Lambda

Amazon RDS, Route 53, Amazon S3, Amazon SES, Amazon SQS et AWS Trusted Advisor Amazon VPC. Pour plus d'informations sur la stratégie gérée, consultez le Guide SystemAdministratorde référence des politiques AWS gérées.

Cette fonction nécessite la capacité de transmettre des rôles aux AWS services. Elle accorde `iam:GetRole` et `iam:PassRole` seulement pour les rôles nommés dans le tableau suivant. Pour plus d'informations, consultez Création des rôles et association des politiques (console) plus loin dans cette rubrique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour des politiques AWS gérées pour les fonctions professionnelles.

Rôles de services IAM facultatifs pour la fonction d'administrateur système

Cas d'utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélection ner	AWS politique gérée à sélection ner
Autoriser les applicati ons s'exécutant dans des instances EC2 d'un cluster Amazon ECS à accéder à Amazon ECS	ecr-sysadmin-*	Rôle Amazon EC2 pour EC2 Container Service	Rôle Amazon EC2 EC2 Container Servicefor
Permettre à un utilisateur de surveiller des bases de données	rds-monitoring-role	Rôle Amazon RDS pour la surveillance améliorée	Rôle Amazon RDS EnhancedM onitoring
Autorisez les applicati ons exécutées dans des instances EC2 à accéder aux AWS ressources.	ec2-sysadmin-*	Amazon EC2	Exemple de politique pour un rôle qui accorde l'accès à un compartiment S3, comme indiqué dans le guide de l'utilisateur Amazon EC2 ; personnal isez-le selon vos besoins

Cas d’utilisation	Nom de rôle (* correspond à un caractère générique)	Type de rôle de service à sélectionner	AWS politique gérée à sélectionner
Autoriser Lambda à lire les flux DynamoDB et à écrire dans les journaux CloudWatch	lambda-sysadmin-*	AWS Lambda	AWSLambdaDynamoDBExecutionRole

Fonction de tâche Utilisateur en affichage seul

AWS nom de la politique gérée : ViewOnly Access

Cas d'utilisation : cet utilisateur peut consulter la liste des AWS ressources et des métadonnées de base du compte pour tous les services. Il ne peut pas lire le contenu des ressources ou les metadonnées au-delà des informations de quotas et de listes des ressources.

Mises à jour de la politique : AWS maintient et met à jour cette politique. Pour obtenir un historique des modifications apportées à cette politique, affichez la politique dans la console IAM, puis choisissez l'onglet Policy versions (Versions de politique. Pour de plus amples informations sur les mises à jour de politique de fonction de tâche, veuillez consulter Mises à jour des politiques AWS gérées pour les fonctions professionnelles.

Description de la politique : Cette politique accorde `List*Describe*`,, `Get*View*`, et `Lookup*` l'accès aux ressources pour les AWS services. Pour connaître les actions incluses dans cette politique pour chaque service, consultez la section ViewOnlyAccès. Pour plus d'informations sur la politique gérée, consultez le Guide de référence ViewOnlyAccess in AWS Managed Policy.

Mises à jour des politiques AWS gérées pour les fonctions professionnelles

Ces politiques sont toutes maintenues AWS et mises à jour pour inclure la prise en charge des nouveaux services et des nouvelles fonctionnalités au fur et à mesure qu'ils sont ajoutés par les AWS services. Elles ne peuvent pas être modifiées par les clients. Vous pouvez créer une copie de la politique, puis la modifier, mais cette copie n'est pas automatiquement mise à jour car elle AWS introduit de nouveaux services et opérations d'API.

Pour une politique de fonction de tâche, vous pouvez afficher l'historique des versions, ainsi que l'heure et la date de chaque mise à jour dans la console IAM. Pour cela, utilisez les liens sur cette

page pour afficher les détails de la politique. Ensuite, choisissez l'onglet Policy versions (Versions de politique) pour afficher les versions. Cette page affiche les 25 dernières versions d'une politique. Pour afficher toutes les versions d'une politique, appelez la AWS CLI commande get-policy-version ou l'opération GetPolicyVersion API.

> **Note**
>
> Vous pouvez avoir jusqu'à cinq versions d'une politique gérée par le client, mais vous pouvez AWS conserver l'historique complet des versions des politiques AWS gérées.

Création des rôles et association des politiques (console)

Plusieurs des politiques répertoriées précédemment permettent de configurer des AWS services avec des rôles qui permettent à ces services d'effectuer des opérations en votre nom. Les politiques de fonctions professionnelles spécifient les noms de rôles exacts à utiliser ou incluent au moins un préfixe qui indique la première partie du nom qui peut être utilisé. Pour créer un de ces rôles, suivez les étapes de la procédure ci-dessous.

Pour créer un rôle pour une Service AWS (console IAM)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.
4. Pour Service ou cas d'utilisation, choisissez un service, puis choisissez le cas d'utilisation. Les cas d'utilisation sont définis par le service pour inclure la politique d'approbation nécessaire au service.
5. Choisissez Suivant.
6. Pour les politiques d'autorisations, les options dépendent du cas d'utilisation que vous avez sélectionné :

   - Si le service définit les autorisations pour le rôle, vous ne pouvez pas sélectionner de politiques d'autorisation.
   - Choisissez parmi un ensemble limité de politiques d'autorisation.
   - Choisissez parmi toutes les politiques d'autorisation.

- Sélectionnez aucune politique d'autorisation, créez les politiques une fois le rôle créé, puis attachez les politiques au rôle.

7. (Facultatif) Définissez une limite d'autorisations. Il s'agit d'une fonctionnalité avancée disponible pour les fonctions de service, mais pas pour les rôles liés à un service.

   a. Ouvrez la section Définir les limites des autorisations, puis choisissez Utiliser une limite d'autorisations pour contrôler le nombre maximal d'autorisations de rôle.

      IAM inclut une liste des politiques AWS gérées et gérées par le client dans votre compte.

   b. Sélectionnez la politique à utiliser comme limite d'autorisations.

8. Choisissez Suivant.
9. Pour le nom du rôle, les options dépendent du service :

   - Si le service définit le nom du rôle, vous ne pouvez pas le modifier.
   - Si le service définit un préfixe pour le nom du rôle, vous pouvez saisir un suffixe facultatif.
   - Si le service ne définit pas le nom du rôle, vous pouvez le nommer.

   > **Important**
   >
   > Lorsque vous nommez un rôle, tenez compte des points suivants :
   >
   > - Les noms de rôles doivent être uniques au sein du Compte AWS vôtre et ne peuvent pas être rendus uniques au cas par cas.
   >
   >   Par exemple, ne créez pas de rôles nommés à la fois **`PRODROLE`** et**`prodrole`**. Lorsqu'un nom de rôle est utilisé dans une politique ou dans le cadre d'un ARN, le nom du rôle distingue les majuscules et minuscules, mais lorsqu'un nom de rôle apparaît aux clients dans la console, par exemple pendant le processus de connexion, le nom du rôle ne distingue pas les majuscules et minuscules.
   > - Vous ne pouvez pas modifier le nom du rôle une fois qu'il a été créé car d'autres entités peuvent y faire référence.

10. (Facultatif) Dans Description, entrez une description pour le rôle.
11. (Facultatif) Pour modifier les cas d'utilisation et les autorisations du rôle, dans les sections Étape 1 : Sélection des entités de confiance ou Étape 2 : Ajouter des autorisations, choisissez Modifier.

12. (Facultatif) Pour identifier, organiser ou rechercher le rôle, ajoutez des balises sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la rubrique Balisage des ressources IAM dans le Guide de l'utilisateur IAM.
13. Passez en revue les informations du rôle, puis choisissez Create role (Créer un rôle).

Exemple 1 : Configuration d'un utilisateur en tant qu'administrateur de base de données (console)

Cet exemple illustre les étapes nécessaires pour configurer Alice, un utilisateur IAM, en tant que Database Administrator (Administrateur de base de données). Vous utilisez les informations dans la première ligne de la table de cette section et autorisez l'utilisateur à activer la surveillance d'Amazon RDS. Vous attachez la DatabaseAdministratorpolitique à l'utilisateur IAM d'Alice afin qu'il puisse gérer les services de base de données Amazon. Cette politique permet également à Alice de transmettre un rôle appelé `rds-monitoring-role` au service Amazon RDS qui autorise ce dernier à superviser les bases de données Amazon RDS en son nom.

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Politiques, tapez **`database`** dans la zone de recherche, puis appuyez sur Entrée.
3. Sélectionnez le bouton radio correspondant à la DatabaseAdministratorpolitique, choisissez Actions, puis choisissez Joindre.
4. Dans la liste des utilisateurs, sélectionnez Alice, puis choisissez Attacher la politique. Alice peut désormais administrer des AWS bases de données. Cependant, pour permettre à Alice de surveiller ces bases de données, vous devez configurer le rôle du service.
5. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
6. Choisissez le type de fonction du AWS service (Service), puis Amazon RDS.
7. Sélectionnez le cas d'utilisation Amazon RDS Role for Enhanced Monitoring.
8. Amazon RDS définit les autorisations pour votre rôle. Choisissez Suivant : Vérification pour continuer.
9. Le nom du rôle doit être l'un de ceux spécifiés par la DatabaseAdministrator politique actuelle d'Alice. L'un de ces noms est **`rds-monitoring-role`**. Saisissez ceci pour le Role name (Nom du rôle).
10. (Facultatif) Dans le champ Description du rôle, saisissez la description du nouveau rôle.
11. Après avoir passé en revue les détails, choisissez Créer un rôle.

12. Alice peut désormais activer RDS Enhanced Monitoring (Surveillance améliorée RDS) dans la section Monitoring (Surveillance) de la console Amazon RDS. Par exemple, elle peut utiliser cette fonction lorsqu'elle crée une instance de base de données ou qu'elle crée un réplica en lecture, ou quand elle modifie une instance de base de données. Ils doivent saisir le nom du rôle qu'ils ont créé (rds-monitoring-role) dans le champ Rôle de surveillance lorsqu'ils définissent Activer la surveillance améliorée sur Oui.

Exemple 2 : Configuration d'un utilisateur en tant qu'administrateur réseau (console)

Cet exemple montre les étapes nécessaires pour configurer Juan, un utilisateur IAM, en tant que Administrateur réseau. Il utilise les informations de la table de cette section pour autoriser Juan à superviser le trafic IP entrant et sortant d'un VPC. Cela permet également à Jorge de capturer ces informations dans les journaux de CloudWatch Logs. Vous attachez la NetworkAdministratorpolitique à l'utilisateur IAM de Jorge afin qu'il puisse configurer les ressources AWS réseau. Cette politique permet également à Juan de transmettre un rôle dont le nom commence par `flow-logs*` à Amazon EC2 lorsque vous créez un journal de flux. Dans ce scénario, à la différence de l'exemple 1, il n'y a aucun type de rôle de service prédéfini. Vous devez donc procéder différemment pour certaines étapes.

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Politiques, puis saisissez **`network`** dans la zone de recherche et appuyez sur Entrée.
3. Sélectionnez le bouton radio à côté de NetworkAdministratorla politique, choisissez Actions, puis choisissez Joindre.
4. Dans la liste des utilisateurs, cochez la case en regard de Juan, puis choisissez Attach policy (Attacher la politique). Jorge peut désormais administrer les ressources AWS du réseau. Cependant, pour activer la surveillance du trafic IP dans votre VPC, vous devez configurer le rôle de service.
5. Comme le rôle de service à créer n'a pas de politique gérée prédéfinie, il doit d'abord être créé. Dans le panneau de navigation, sélectionnez Politicies (Politiques), puis Create policy (Créer une politique).
6. Dans la section Éditeur de politiques, choisissez l'option JSON et copiez le texte du document de stratégie JSON suivant. Collez ce texte dans la zone de texte JSON.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

7. Résolvez les avertissements de sécurité, les erreurs ou les avertissements généraux générés durant la validation de la politique, puis choisissez Suivant.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez Restructuration de politique.

8. Sur la page Vérifier et créer, tapez **`vpc-flow-logs-policy-for-service-role`** pour le nom de la politique. Vérifiez les Autorisations définies dans cette politique pour voir les autorisations accordées par votre politique, puis choisissez Créer une politique pour enregistrer votre travail.

   La nouvelle politique s'affiche dans la liste des politiques gérées et est prête à être attachée.

9. Dans le panneau de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
10. Choisissez le type de fonction du AWS service (Service), puis Amazon EC2.
11. Sélectionnez le cas d'utilisation Amazon EC2.
12. Sur la page Joindre des politiques d'autorisation, choisissez la politique que vous avez créée précédemmentfor-service-role, vpc-flow-logs-policy-, puis choisissez Suivant : Réviser.

13. Le nom du rôle doit être autorisé par la NetworkAdministrator politique actuelle de Jorge. Tout nom qui commence par `flow-logs-` est autorisé. Pour cet exemple, saisissez **`flow-logs-for-jorge`** comme Role name (Nom du rôle).
14. (Facultatif) Dans le champ Description du rôle, saisissez la description du nouveau rôle.
15. Après avoir passé en revue les détails, choisissez Créer un rôle.
16. Vous pouvez désormais configurer la politique d'approbation nécessaire pour ce scénario. Sur la page Rôles, choisissez le flow-logs-for-jorgerôle (le nom, pas la case à cocher). Sur la page des détails de votre nouveau rôle, choisissez l'onglet Relations d'approbation, puis choisissez Modifier la relation d'approbation.
17. Modifier la ligne « Service » comme suit, en remplaçant l'entrée pour `ec2.amazonaws.com` :

    ```
            "Service": "vpc-flow-logs.amazonaws.com"
    ```

18. Juan peut désormais créer des journaux de flux pour un VPC ou un sous-réseau dans la console Amazon EC2. Lorsque vous créez le journal de flux, spécifiez le flow-logs-for-jorgerôle. Ce rôle dispose des autorisations pour créer le journal et y consigner des données.

AWS clés contextuelles de condition globale

Lorsqu'un principal fait une demande à AWS, AWS rassemble les informations de la demande dans un contexte de demande. Vous pouvez utiliser l'élément `Condition` d'une politique JSON pour comparer des clés dans le contexte de demande avec les valeurs de clé spécifiées dans votre politique. Les informations relatives à la demande sont fournies par différentes sources, notamment le principal auteur de la demande, la ressource visée par la demande et les métadonnées relatives à la demande elle-même.

Les clés de condition globales peuvent être utilisées dans tous les AWS services. Bien que ces clés de condition puissent être utilisées dans toutes les politiques, elles ne sont pas disponibles dans tous les contextes de demande. Par exemple, la clé de `aws:SourceAccount` condition n'est disponible que lorsque l'appel à votre ressource est effectué directement par un principal AWS de service. Pour en savoir plus sur les circonstances dans lesquelles une clé globale est incluse dans le contexte de la demande, consultez les informations de disponibilité pour chaque clé.

Certains services individuels créent leurs propres clés de condition qui sont disponibles dans le contexte de la demande pour d'autres services. Les clés de condition interservices sont un type de clé de condition globale qui inclut un préfixe correspondant au nom du service, tel que `ec2:` ou`lambda:`, mais qui sont disponibles dans d'autres services.

Les clés de condition spécifiques au service sont définies pour être utilisées avec un service individuel AWS . Par exemple, Amazon S3 vous permet de rédiger une politique avec la clé de `s3:VersionId` condition pour limiter l'accès à une version spécifique d'un objet Amazon S3. Cette clé de condition est propre au service, ce qui signifie qu'elle ne fonctionne qu'avec les demandes adressées au service Amazon S3. Pour les clés de condition spécifiques à un service, consultez Actions, ressources et clés de condition pour les AWS services et choisissez le service dont vous souhaitez afficher les clés.

> **Note**
>
> Si vous utilisez des clés de condition qui ne sont disponibles que dans certaines circonstances, vous pouvez utiliser les IfExistsversions des opérateurs de condition. Si les clés de condition ne figurent pas dans le contexte de la demande, l'évaluation de la politique peut échouer. Par exemple, utilisez le bloc de condition suivant avec les opérateurs `...IfExists` pour déterminer quand la demande provient d'une plage d'adresses IP ou d'un VPC spécifique. Si l'une des clés, ou les deux, ne figurent pas dans le contexte de la demande, la condition renvoie toujours `true`. Les valeurs ne sont vérifiées que si la clé spécifiée figure dans le contexte de la demande. Pour plus d'informations sur la manière dont une politique est évaluée lorsqu'une clé n'est pas présente pour les autres opérateurs, consultez la section Opérateurs de condition.
>
> ```
> "Condition": {
>     "IpAddressIfExists": {"aws:SourceIp" : ["xxx"] },
>     "StringEqualsIfExists" : {"aws:SourceVpc" : ["yyy"]}
> }
> ```

> **Important**
>
> Pour comparer votre condition à un contexte de demande avec plusieurs valeurs de clé, vous devez utiliser les opérateurs d'ensemble `ForAllValues` ou `ForAnyValue`. Utilisez des opérateurs d'ensemble uniquement avec des clés de condition à valeurs multiples. N'utilisez pas d'opérateurs d'ensemble avec des clés de condition à valeur unique. Pour plus d'informations, consultez Clés de contexte à valeurs multiples.

Propriétés du principal	Propriétés d'une session de rôle	Propriétés du réseau	Propriétés de la ressource	Propriétés de la demande
lois : PrincipalArn lois : PrincipalAccount aws : PrincipalOrg Chemins lois : PrincipalOrg ID aws :PrincipalTag//tag-key lois : PrincipalIsAWSService aws : PrincipalService Nom lois : PrincipalService NamesList lois : PrincipalType aws:userid aws:username	lois : FederatedProvider lois : TokenIssueHeure lois : MultiFactorAuthAge lois : MultiFactorAuthPresent AWS : EC2 VpcInstanceSource AWS : EC2PrivateIPv4InstanceSource lois : SourceIdentity EC2 : RoleDelivery ec2 : ArneSourceInstance colle : RoleAssumedPar colle : CredentialIssuing Service lambda : ArnSourceFunction	lois : SourceIp lois : SourceVpc lois : SourceVpce lois : VpcSourceIP	lois : ResourceAccount aws : ResourceOrg Chemins lois : ResourceOrg ID aws :ResourceTag//tag-key	lois : CalledVia aws : CalledViaTout d'abord aws : CalledViaDernier AWS : via AWSService lois : CurrentTime lois : EpochTime aws:referer lois : RequestedRegion aws :RequestTag//tag-key lois : TagKeys lois : SecureTransport lois : SourceArn lois : SourceAccount aws : SourceOrgChemins

Propriétés du principal	Propriétés d'une session de rôle	Propriétés du réseau	Propriétés de la ressource	Propriétés de la demande
	ssm : bras SourceInstance boutique d'identité : UserId			lois : SourceOrg ID lois : UserAgent

Propriétés du principal

Utilisez les clés de condition suivantes pour comparer les détails concernant le principal auteur de la demande avec les propriétés principales que vous spécifiez dans la politique. Pour obtenir la liste des principaux habilités à faire des demandes, consultezSpécification d'un principal.

Table des matières

- lois : PrincipalArn
- lois : PrincipalAccount
- aws : PrincipalOrg Chemins
- lois : PrincipalOrg ID
- aws :PrincipalTag//tag-key
- lois : PrincipalIs AWSService
- aws : PrincipalService Nom
- lois : PrincipalService NamesList
- lois : PrincipalType
- aws:userid
- aws:username

lois : PrincipalArn

Utilisez cette clé pour comparer l'Amazon Resource Name (ARN) du principal ayant fait la demande avec l'ARN spécifié dans la politique. Pour les rôles IAM, le contexte de la demande renvoie l'ARN du rôle, et non l'ARN de l'utilisateur qui a endossé le rôle.

- Availability (Disponibilité) : cette clé figure dans le contexte de la demande pour toutes les demandes signées. Les demandes anonymes n'incluent pas cette clé. Vous pouvez spécifier les types de mandataires suivants dans cette clé de condition :
  - Rôle IAM
  - Utilisateur IAM
  - AWS STS session utilisateur fédérée
  - Compte AWS utilisateur root
- Type de données : ARN, chaîne

  AWS recommande d'utiliser des opérateurs ARN plutôt que des opérateurs de chaîne lorsque vous comparez des ARN.
- Type de valeur – À valeur unique
- Exemples de valeurs La liste suivante indique la valeur de contexte de demande renvoyée pour différents types de principes que vous pouvez spécifier dans la clé de `aws:PrincipalArn` condition :
  - Rôle IAM— Le contexte de la requête contient la valeur suivante pour la clé de condition `aws:PrincipalArn`. Ne spécifiez pas l'ARN de session de rôle présumé comme valeur pour cette clé de condition. Pour en savoir plus sur le rôle présumé de session principale, reportez-vous à la section Principaux de séance de rôle.

    ```
    arn:aws:iam::123456789012:role/role-name
    ```
  - Utilisateur IAM— Le contexte de la requête contient la valeur suivante pour la clé de conditionaws:PrincipalArn.

    ```
    arn:aws:iam::123456789012:user/user-name
    ```
  - AWS STS sessions utilisateur fédérées : le contexte de demande contient la valeur suivante pour la clé `aws:PrincipalArn` de condition.

    ```
    arn:aws:sts::123456789012:federated-user/user-name
    ```
  - Compte AWS utilisateur root — Le contexte de demande contient la valeur suivante pour la clé de conditionaws:PrincipalArn. Lorsque vous spécifiez l'ARN de l'utilisateur root comme valeur pour la `aws:PrincipalArn` clé de condition, il ne limite les autorisations que pour l'utilisateur root de Compte AWS. Cela diffère de la spécification de l'ARN de l'utilisateur root dans l'élément principal d'une politique basée sur les ressources, qui délègue l'autorité au

Compte AWS. Pour en savoir plus sur la spécification de l'ARN de l'utilisateur root dans l'élément principal d'une politique basée sur les ressources, consultez Compte AWS principes.

```
arn:aws:iam::123456789012:root
```

Vous pouvez spécifier l'ARN de l'utilisateur root comme valeur de clé de condition `aws:PrincipalArn` dans les politiques AWS Organizations de contrôle des services (SCP). Les SCP sont un type de stratégie d'organisation que vous pouvez utiliser pour gérer les autorisations dans votre organisation ; elles n'affectent que les comptes des membres dans l'organisation. Une SCP limite les autorisations des utilisateurs et des rôles IAM dans les comptes membres, y compris l'utilisateur racine du compte membre. Pour plus d'informations sur l'effet des SCP sur les autorisations, consultez les effets des SCP sur les autorisations dans le Guide d'utilisation pour les organisations.

lois : PrincipalAccount

Utilisez cette clé pour comparer le compte auquel appartient le principal demandeur avec l'identifiant de compte spécifié dans la politique. Pour les demandes anonymes, le contexte de la demande renvoie `anonymous`.

- Disponibilité : cette clé figure dans le contexte de la demande pour toutes les demandes, y compris les requêtes anonymes.
- Type de données : chaîne
- Type de valeur – À valeur unique

Dans l'exemple suivant, l'accès est refusé, sauf aux principaux possédant le numéro de compte `123456789012`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAccessFromPrincipalNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:region:accountID:resource"
      ],
```

```
      "Condition": {
        "StringNotEquals": {
          "aws:PrincipalAccount": [
            "123456789012"
          ]
        }
      }
    }
  ]
}
```

aws : PrincipalOrg Chemins

Utilisez cette clé pour comparer le AWS Organizations chemin du principal qui fait la demande avec le chemin indiqué dans la politique. Ce principal peut être un utilisateur IAM, un rôle IAM, un utilisateur fédéré ou. Utilisateur racine d'un compte AWS Dans une politique, cette clé de condition vérifie que le demandeur est un membre du compte au sein de l'organisation racine ou des unités d'organisation spécifiées dans AWS Organizations. Un AWS Organizations chemin est une représentation textuelle de la structure d'une entité Organizations. Pour de plus amples informations sur l'utilisation et la compréhension des chemins, veuillez consulter Comprendre le chemin d'entité AWS Organizations.

- Availability (Disponibilité) : cette clé ne figure dans le contexte de la demande que si le principal est membre d'une organisation. Les demandes anonymes n'incluent pas cette clé.
- Type de données — Chaîne (liste)
- Type de valeur – À valeur multiple

> **Note**
>
> Les ID d'organisation sont globalement uniques, mais les ID d'unité d'organisation et les ID racine ne sont uniques qu'au sein d'une organisation. Cela signifie qu'aucune organisation ne partage le même ID d'organisation. Toutefois, une autre organisation peut avoir le même ID d'unité d'organisation ou ID racine que vous. Nous vous recommandons de toujours inclure l'ID d'organisation lorsque vous spécifiez une unité d'organisation ou une racine.

Par exemple, la condition suivante renvoie `true` pour les principaux de comptes qui sont directement attachés à l'unité d'organisation `ou-ab12-22222222`, et non à ses unités d'organisation enfants.

```
"Condition" : { "ForAnyValue:StringEquals" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

La condition suivante renvoie `true` pour les principaux d'un compte qui est directement attaché à l'unité d'organisation ou à l'une de ses unités d'organisation enfants. Lorsque vous incluez un caractère générique, vous devez utiliser l'opérateur de condition `StringLike`.

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
*"]
}}
```

La condition suivante renvoie `true` pour les principaux d'un compte qui est directement attaché à l'une des unités d'organisation enfant, mais pas directement à l'unité d'organisation parent. La condition précédente s'applique à l'unité d'organisation ou à tous les enfants. La condition suivante s'applique uniquement aux enfants (et aux enfants de ces enfants).

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
ou-*"]
}}
```

La condition suivante autorise l'accès à chaque principal de l'organisation `o-a1b2c3d4e5`, quelle que soit l'unité d'organisation parent.

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/*"]
}}
```

`aws:PrincipalOrgPaths` est une clé de condition à valeurs multiples. Les clés multivaluées peuvent avoir plusieurs valeurs dans le contexte de la demande. Lorsque vous utilisez plusieurs valeurs avec l'opérateur de condition `ForAnyValue`, le chemin d'accès du principal doit correspondre à l'un des chemins spécifiés dans la politique. Pour de plus amples informations sur les clés de condition à valeurs multiples, veuillez consulter Clés de contexte à valeurs multiples.

```
    "Condition": {
        "ForAnyValue:StringLike": {
            "aws:PrincipalOrgPaths": [
                "o-a1b2c3d4e5/r-ab12/ou-ab12-33333333/*",
                "o-a1b2c3d4e5/r-ab12/ou-ab12-22222222/*"
            ]
        }
    }
```

lois : PrincipalOrg ID

Utilisez cette clé pour comparer l'identifiant de l'organisation AWS Organizations à laquelle appartient le principal demandeur avec l'identifiant spécifié dans la politique.

- Availability (Disponibilité) : cette clé ne figure dans le contexte de la demande que si le principal est membre d'une organisation. Les demandes anonymes n'incluent pas cette clé.
- Type de données : chaîne
- Type de valeur – À valeur unique

Cette clé globale permet d'éviter de répertorier tous les ID de compte pour tous les comptes AWS d'une organisation. Vous pouvez utiliser cette clé de condition pour simplifier la spécification de l'élément `Principal` d'une politique basée sur une ressource. Vous pouvez spécifier l'ID de l'organisation dans l'élément Condition. Lorsque vous ajoutez et supprimez des comptes, les politiques qui contiennent la clé `aws:PrincipalOrgID` ajoutent automatiquement les bons comptes et ne nécessitent pas de mise à jour manuelle.

Par exemple, la politique de compartiment Amazon S3 suivante permet aux membres de n'importe quel compte de l'organisation `o-xxxxxxxxxxx` d'ajouter un objet dans le compartiment `policy-ninja-dev`.

```
{
 "Version": "2012-10-17",
 "Statement": {
   "Sid": "AllowPutObject",
   "Effect": "Allow",
   "Principal": "*",
   "Action": "s3:PutObject",
   "Resource": "arn:aws:s3:::policy-ninja-dev/*",
   "Condition": {"StringEquals":
```

```
      {"aws:PrincipalOrgID":"o-xxxxxxxxxxx"}
    }
  }
}
```

> **Note**
>
> Cette condition globale s'applique également au compte de gestion d'une organisation AWS . Cette politique empêche tous les principaux en dehors de l'organisation spécifiée d'accéder au compartiment Amazon S3. Cela inclut tous AWS les services qui interagissent avec vos ressources internes, tels que AWS CloudTrail l'envoi de données de journal vers vos compartiments Amazon S3. Pour savoir comment accorder l'accès aux AWS services en toute sécurité, consultezlois : PrincipalIs AWSService.

Pour plus d'informations AWS Organizations, voir Qu'est-ce que c'est AWS Organizations ? dans le guide de AWS Organizations l'utilisateur.

aws :PrincipalTag//tag-key

Utilisez cette clé pour comparer la balise attachée au principal effectuant la demande avec la balise spécifiée dans la politique. Si plusieurs balises sont attachées au principal, le contexte de la demande inclut une clé `aws:PrincipalTag` pour chaque clé de balise attachée.

- Availability (Disponibilité) : cette clé figure dans le contexte de la demande si le principal utilise un utilisateur IAM avec des balises attachées. Elle est incluse pour un principal utilisant un rôle IAM avec des balises attachées ou des balises de session. Les demandes anonymes n'incluent pas cette clé.
- Type de données : chaîne
- Type de valeur – À valeur unique

Vous pouvez ajouter des attributs personnalisés à un utilisateur ou un rôle sous la forme d'une paire clé-valeur. Pour plus d'informations sur les balises IAM, consultez Balisage des ressources IAM. Vous pouvez utiliser `aws:PrincipalTag`pour le contrôle d'accès des principaux AWS .

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise les utilisateurs avec la balise **`department=hr`** à gérer les utilisateurs, groupes ou rôles IAM. Pour utiliser cette politique, remplacez le *`texte de l'espace réservé en italique`* dans l'exemple

de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": "hr"
        }
      }
    }
  ]
}
```

lois : PrincipalIs AWSService

Utilisez cette touche pour vérifier si l'appel à votre ressource est effectué directement par un directeur AWS de service. Par exemple, AWS CloudTrail utilise le principal de service `cloudtrail.amazonaws.com` pour écrire des journaux dans votre compartiment Amazon S3. La clé de contexte de demande est définie sur true lorsqu'un service utilise un principal de service pour effectuer une action directe sur vos ressources. La clé de contexte est définie sur false si le service utilise les informations d'identification d'un principal IAM pour effectuer une demande au nom du principal. Elle est également définie sur false si le service utilise un rôle de service ou un rôle lié à un service pour effectuer un appel au nom du principal.

- Disponibilité : cette clé est présente dans le contexte de la demande pour toutes les demandes d'API signées utilisant des informations d'identification AWS . Les demandes anonymes n'incluent pas cette clé.
- Type de données : booléen
- Type de valeur – À valeur unique

Vous pouvez utiliser cette clé de condition pour limiter l'accès à vos identités fiables et aux emplacements réseau attendus tout en accordant l'accès aux AWS services en toute sécurité.

Dans l'exemple de politique de compartiment Amazon S3 suivant, l'accès au compartiment est restreint sauf si la demande provient `vpc-111bbb22` ou provient d'un principal de service, tel que CloudTrail.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Expected-network+service-principal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/AWS Logs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22"
        },
        "BoolIfExists": {
          "aws:PrincipalIsAWSService": "false"
        }
      }
    }
  ]
}
```

Dans la vidéo suivante, découvrez comment utiliser la clé de condition `aws:PrincipalIsAWSService` dans une politique.

Accordez l'accès en toute sécurité à vos utilisateurs autorisés, aux emplacements réseau attendus et AWS aux services en même temps.

aws : PrincipalService Nom

Utilisez cette clé pour comparer le nom du principal de service dans la politique au principal de service qui effectue des demandes à vos ressources. Vous pouvez utiliser cette clé pour vérifier si cet appel est effectué par un principal de service spécifique. Lorsqu'un principal de service effectue une demande directe à vos ressources, la clé `aws:PrincipalServiceName` contient le nom du principal de service. Par exemple, le nom principal du AWS CloudTrail service est`cloudtrail.amazonaws.com`.

- Disponibilité — Cette clé est présente dans la demande lorsque l'appel est effectué par un directeur AWS de service. Cette clé n'est présente dans aucun autre cas, notamment les suivants :

  - Si le service utilise un rôle de service ou un rôle lié à un service pour effectuer un appel au nom du principal.
  - Si le service utilise les informations d'identification d'un principal IAM pour effectuer une demande au nom du principal.
  - Si l'appel est effectué directement par un principal IAM.
  - Si l'appel est passé par un demandeur anonyme.
- Type de données : chaîne
- Type de valeur – À valeur unique

Vous pouvez utiliser cette clé de condition pour limiter l'accès à vos identités fiables et aux emplacements réseau attendus tout en accordant l'accès à un AWS service en toute sécurité.

Dans l'exemple de politique de compartiment Amazon S3 suivant, l'accès au compartiment est restreint sauf si la demande provient `vpc-111bbb22` ou provient d'un principal de service, tel que CloudTrail.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "expected-network+service-principal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/AWS Logs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22",
          "aws:PrincipalServiceName": "cloudtrail.amazonaws.com"
        }
      }
    }
  ]
}
```

lois : PrincipalService NamesList

Cette clé fournit une liste de tous les noms des principal de service appartenant au service. Il s'agit d'une clé de condition avancée. Vous pouvez l'utiliser pour empêcher le service d'accéder à vos ressources à partir d'une région spécifique uniquement. Certains services peuvent créer des principaux de services régionaux pour indiquer une instance particulière du service dans une région spécifique. Vous pouvez limiter l'accès à une ressource à une instance particulière du service. Lorsqu'un principal de service effectue une demande directe à vos ressources, `aws:PrincipalServiceNamesList` contient une liste non ordonnée de tous les noms des principaux de services associés à l'instance régionale du service.

- Disponibilité — Cette clé est présente dans la demande lorsque l'appel est effectué par un directeur AWS de service. Cette clé n'est présente dans aucun autre cas, notamment les suivants :
  - Si le service utilise un rôle de service ou un rôle lié à un service pour effectuer un appel au nom du principal.
  - Si le service utilise les informations d'identification d'un principal IAM pour effectuer une demande au nom du principal.
  - Si l'appel est effectué directement par un principal IAM.
  - Si l'appel est passé par un demandeur anonyme.
- Type de données — Chaîne (liste)
- Type de valeur – À valeur multiple

`aws:PrincipalServiceNamesList` est une clé de condition à valeurs multiples. Les clés multivaluées peuvent avoir plusieurs valeurs dans le contexte de la demande. Vous devez utiliser les opérateurs d'ensemble `ForAnyValue` ou `ForAllValues` avec l'opérateur de condition de chaîne pour cette clé. Pour de plus amples informations sur les clés de condition à valeurs multiples, veuillez consulter Clés de contexte à valeurs multiples.

lois : PrincipalType

Utilisez cette clé pour comparer le type de principal qui effectue la demande avec le type de principal spécifié dans la politique. Pour plus d'informations, veuillez consulter Spécification d'un principal. Pour des exemples spécifiques de valeurs clés `principal`, veuillez consulter Valeurs de la clé du principal.

- Disponibilité : cette clé figure dans le contexte de la demande pour toutes les demandes, y compris les requêtes anonymes.

- Type de données : chaîne
- Type de valeur – À valeur unique

aws:userid

Utilisez cette clé pour comparer l'identifiant principal du demandeur avec l'ID spécifié dans la politique. Pour les utilisateurs IAM, la valeur du contexte de la demande est l'ID utilisateur. Pour les rôles IAM, ce format de valeur peut varier. Pour de plus amples informations sur l'ajout de principaux, veuillez consulter Spécification d'un principal. Pour des exemples spécifiques de valeurs clés `principal`, veuillez consulter Valeurs de la clé du principal.

- Disponibilité : cette clé figure dans le contexte de la demande pour toutes les demandes, y compris les requêtes anonymes.
- Type de données : chaîne
- Type de valeur – À valeur unique

aws:username

Utilisez cette clé pour comparer le nom d'utilisateur du demandeur avec celui spécifié dans la politique. Pour de plus amples informations sur l'ajout de principaux, veuillez consulter Spécification d'un principal. Pour des exemples spécifiques de valeurs clés `principal`, veuillez consulter Valeurs de la clé du principal.

- Availability (Disponibilité) : cette clé figure toujours dans le contexte de la demande pour les utilisateurs IAM. Les demandes anonymes et les demandes effectuées à l'aide des rôles Utilisateur racine d'un compte AWS ou IAM n'incluent pas cette clé. Les requêtes effectuées à l'aide des informations d'identification IAM Identity Center n'incluent pas cette clé dans le contexte.
- Type de données : chaîne
- Type de valeur – À valeur unique

Propriétés d'une session de rôle

Utilisez les clés de condition suivantes pour comparer les propriétés de la session de rôle au moment où la session a été générée. Ces clés de condition ne sont disponibles que lorsqu'une demande est faite par un principal doté d'une session de rôle ou d'informations d'identification utilisateur fédérées. Les valeurs de ces clés de condition sont intégrées dans le jeton de session du rôle.

Un rôle est une sorte de principal. Vous pouvez également utiliser les clés de condition de la Propriétés du principal section pour évaluer les propriétés d'un rôle lorsqu'un rôle fait une demande.

Table des matières

- lois : FederatedProvider
- lois : TokenIssue Heure
- lois : MultiFactor AuthAge
- lois : MultiFactor AuthPresent
- AWS : EC2 Vpc InstanceSource
- AWS : EC2 PrivateIPv4 InstanceSource
- lois : SourceIdentity
- EC2 : RoleDelivery
- ec2 : Arne SourceInstance
- colle : RoleAssumed Par
- colle : CredentialIssuing Service
- lambda : Arn SourceFunction
- ssm : bras SourceInstance
- boutique d'identité : UserId

lois : FederatedProvider

Utilisez cette clé pour comparer le fournisseur d'identité (IdP) émetteur du principal avec l'IdP que vous spécifiez dans la politique. Cela signifie qu'un rôle IAM a été assumé à l'aide de l'`AssumeRoleWithWebIdentity` AWS STS opération. Lorsque les informations d'identification temporaires de la séance de rôle résultante sont utilisées pour effectuer une demande, le contexte de la demande identifie l'IdP qui a authentifié l'identité fédérée d'origine.

- Disponibilité : cette clé est présente lorsque le principal est un principal de séance de rôle et que cette séance a été émise lorsqu'un rôle a été assumé avec `AssumeRoleWithWebIdentity`.
- Type de données : chaîne
- Type de valeur – À valeur unique

Par exemple, si l'utilisateur s'est authentifié via Amazon Cognito, le contexte de la demande inclut la valeur `cognito-identity.amazonaws.com`. De la même façon, si l'utilisateur s'est authentifié via Login with Amazon, le contexte de la demande inclut la valeur `www.amazon.com`.

Vous pouvez utiliser n'importe quelle clé de condition à valeur unique comme variable. L'exemple suivant de politique basée sur les ressources utilise la clé `aws:FederatedProvider` comme variable de politique dans l'ARN d'une ressource. Cette politique permet à tout principal qui s'est authentifié à l'aide d'un IdP d'obtenir des objets à partir d'un compartiment Amazon S3 avec un chemin spécifique au fournisseur d'identité émetteur.

lois : TokenIssue Heure

Utilisez cette clé pour comparer la date et l'heure d'émission des informations d'identification de sécurité temporaires avec celles spécifiées dans la politique.

- Availability (Disponibilité) : cette clé figure dans le contexte de la demande uniquement lorsque le principal utilise des informations d'identification temporaires pour effectuer la demande. La clé n'est pas présente dans AWS CLI les demandes AWS d'API ou de AWS SDK effectuées à l'aide de clés d'accès.
- Type de données — Date
- Type de valeur – À valeur unique

Pour savoir quels services prennent en charge l'utilisation d'informations d'identification temporaires, consultez AWS services qui fonctionnent avec IAM.

lois : MultiFactor AuthAge

Utilisez cette clé pour comparer le nombre de secondes écoulées depuis l'authentification du principal demandeur avec MFA avec le nombre spécifié dans la politique. Pour plus d'informations sur l'authentification MFA, consultez Utilisation de l'authentification multifactorielle (MFA) dans AWS.

> **Important**
>
> Cette clé de condition n'est pas présente pour les identités fédérées ou les demandes effectuées à l'aide de clés d'accès pour signer les demandes de AWS CLI, d' AWS API ou de AWS SDK. Pour en savoir plus sur l'ajout d'une protection MFA aux opérations d'API à l'aide d'informations d'identification de sécurité temporaires, consultez. Configuration de l'accès aux API protégé par MFA

Pour vérifier si le MFA est utilisé pour valider les identités fédérées IAM, vous pouvez transmettre la méthode d'authentification de votre fournisseur d'identité à AWS une balise de session. Pour plus de détails, consultez Transmission des balises de session AWS STS. Pour appliquer le MFA aux identités IAM Identity Center, vous pouvez activer les attributs de contrôle d'accès afin de transmettre une demande d'assertion SAML avec la méthode d'authentification de votre fournisseur d'identité à IAM Identity Center.

- Disponibilité — Cette clé est incluse dans le contexte de la demande uniquement lorsque le principal utilise des informations d'identification de sécurité temporaires pour effectuer la demande. Les politiques assorties de conditions MFA peuvent être jointes à :
  - Un utilisateur ou un groupe IAM
  - Une ressource telle qu'un compartiment Amazon S3, une file d'attente Amazon SQS ou une rubrique Amazon SNS
  - La politique d'approbation d'un rôle IAM qui peut être endossée par un utilisateur
- Type de données : numérique
- Type de valeur – À valeur unique

lois : MultiFactor AuthPresent

Utilisez cette clé pour vérifier si l'authentification multifactorielle (MFA) a été utilisée pour valider les informations de sécurité temporaires à l'origine de la demande.

⚠ Important

Cette clé de condition n'est pas présente pour les identités fédérées ou les demandes effectuées à l'aide de clés d'accès pour signer les demandes de AWS CLI, d' AWS API ou de AWS SDK. Pour en savoir plus sur l'ajout d'une protection MFA aux opérations d'API à l'aide d'informations d'identification de sécurité temporaires, consultez. Configuration de l'accès aux API protégé par MFA

Pour vérifier si le MFA est utilisé pour valider les identités fédérées IAM, vous pouvez transmettre la méthode d'authentification de votre fournisseur d'identité à AWS une balise de session. Pour plus de détails, consultez Transmission des balises de session AWS STS. Pour appliquer le MFA aux identités IAM Identity Center, vous pouvez activer les attributs

de contrôle d'accès afin de transmettre une demande d'assertion SAML avec la méthode d'authentification de votre fournisseur d'identité à IAM Identity Center.

- Availability (Disponibilité) : cette clé figure dans le contexte de la demande uniquement lorsque le principal utilise des informations d'identification temporaires pour effectuer la demande. Les politiques assorties de conditions MFA peuvent être jointes à :
  - Un utilisateur ou un groupe IAM
  - Une ressource telle qu'un compartiment Amazon S3, une file d'attente Amazon SQS ou une rubrique Amazon SNS
  - La politique d'approbation d'un rôle IAM qui peut être endossée par un utilisateur
- Type de données : booléen
- Type de valeur – À valeur unique

Les informations d'identification temporaires sont utilisées pour authentifier les rôles IAM et les utilisateurs IAM à l'aide de jetons temporaires provenant de AssumeRoleou de GetSessionjetons, ainsi que les utilisateurs du. AWS Management Console

Les clés d'accès utilisateur IAM sont des informations d'identification à long terme, mais dans certains cas, elles AWS créent des informations d'identification temporaires au nom des utilisateurs IAM pour effectuer des opérations. Dans ces cas, la clé `aws:MultiFactorAuthPresent` est présente dans la demande et définie sur la valeur `false`. Deux scénarios courants peuvent expliquer ce comportement :

- Les utilisateurs IAM utilisent AWS Management Console sans le savoir des informations d'identification temporaires. Les utilisateurs se connectent à la console avec leur nom d'utilisateur et leur mot de passe, qui sont des informations d'identification à long terme. Toutefois, en arrière-plan, la console génère des informations d'identification temporaires pour le compte de l'utilisateur.
- Si un utilisateur IAM appelle un AWS service, le service réutilise les informations d'identification de l'utilisateur pour faire une autre demande à un autre service. Par exemple, lorsque vous appelez Athena pour accéder à un compartiment Amazon S3 ou lorsque vous l'utilisez AWS CloudFormation pour créer une instance Amazon EC2. Pour la demande suivante, AWS utilise des informations d'identification temporaires.

Pour savoir quels services prennent en charge l'utilisation d'informations d'identification temporaires, consultez AWS services qui fonctionnent avec IAM.

La clé `aws:MultiFactorAuthPresent` n'est jamais présente lorsqu'une API ou une commande de l'interface de ligne de commande (CLI) est appelée avec des informations d'identification à long terme, telles que des paires de clés d'accès. Par conséquent, nous vous recommandons d'utiliser les versions `...IfExists` des opérateurs de condition lors de la vérification de cette clé.

Il est important de comprendre que l'élément `Condition` suivant ne constitue pas une méthode fiable pour vérifier si une demande est authentifiée avec MFA.

```
#####   WARNING: NOT RECOMMENDED   #####
"Effect" : "Deny",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Cette combinaison de l'effet `Deny`, de l'élément `Bool` et de la valeur `false` refuse les demandes qui peuvent être authentifiées à l'aide de MFA, mais ne l'ont pas été. Cela s'applique uniquement aux informations d'identification temporaires qui prennent en charge l'utilisation de l'authentification MFA. Cette instruction ne refuse pas l'accès aux demandes effectuées à l'aide d'informations d'identification à long terme, ni aux demandes qui ont été authentifiées avec MFA. Utilisez cet exemple avec précaution, car sa logique est complexe et qu'il ne teste pas si l'authentification MFA a été effectivement utilisée.

De même, n'utilisez pas la combinaison de l'effet `Deny`, de l'élément `Null` et de `true`, car elle se comporte de la même manière et sa logique est encore plus complexe.

Combinaison recommandée

Nous vous recommandons plutôt d'utiliser l'opérateur `BoolIfExists` pour vérifier si une demande est authentifiée à l'aide de MFA.

```
"Effect" : "Deny",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Cette combinaison de `Deny`, `BoolIfExists` et `false` refuse les demandes qui ne sont pas authentifiées à l'aide de MFA. Elle refuse en particulier les demandes provenant d'informations d'identification temporaires qui n'incluent pas l'authentification MFA. Il refuse également les demandes effectuées à l'aide d'informations d'identification à long terme, telles que les opérations AWS CLI d' AWS API effectuées à l'aide de clés d'accès. L'opérateur `*IfExists` vérifie si la clé `aws:MultiFactorAuthPresent` existe et est présente ou non. Utilisez cet opérateur si vous

souhaitez refuser une demande qui n'est pas authentifiée à l'aide de MFA. Ceci est plus sûr, mais peut casser tout code ou script utilisant des clés d'accès pour accéder à l' AWS API AWS CLI or.

Combinaisons alternatives

Vous pouvez également utiliser l'`BoolIfExists`opérateur pour autoriser les demandes authentifiées par MFA et/ou les demandes d' AWS API effectuées à l' AWS CLI aide d'informations d'identification à long terme.

```
"Effect" : "Allow",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "true" } }
```

Cette condition correspond à deux cas : si la clé existe et est présente ou si la clé n'existe pas. Cette combinaison de `Allow`, `BoolIfExists` et `true` autorise les demandes qui sont authentifiées à l'aide de MFA, ou les demandes qui ne peuvent pas être authentifiées à l'aide de MFA. Cela signifie que AWS CLI les opérations AWS d'API et de AWS SDK sont autorisées lorsque le demandeur utilise ses clés d'accès à long terme. Cette combinaison n'autorise pas les demandes provenant d'informations d'identification temporaires qui pourraient, mais n'incluent pas MFA.

Lorsque vous créez une politique à l'aide de l'éditeur visuel de la console IAM et que vous sélectionnez MFA required (MFA obligatoire), cette combinaison s'applique. Ce paramètre nécessite l'authentification MFA pour l'accès à la console, mais autorise un accès par programmation sans authentification MFA.

Sinon, vous pouvez utiliser l'opérateur `Bool` pour autoriser les demandes par programmation et les demandes de console uniquement lorsqu'elles sont authentifiées à l'aide de MFA.

```
"Effect" : "Allow",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "true" } }
```

Cette combinaison de `Allow`, `Bool` et `true` autorise uniquement les demandes authentifiées par MFA. Cela s'applique uniquement aux informations d'identification temporaires qui prennent en charge l'utilisation de l'authentification MFA. Cette instruction n'autorise pas l'accès aux demandes effectuées à l'aide de clés d'accès à long terme ou d'informations d'identification temporaires sans authentification MFA.

N'utilisez pas de construction de politique semblable à la suivante pour vérifier la présence de la clé MFA :

```
#####   WARNING: USE WITH CAUTION   #####
```

```
"Effect" : "Allow",
"Condition" : { "Null" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Cette combinaison de l'effet `Allow`, de l'élément `Null` et de la valeur `false` autorise uniquement les demandes qui peuvent être authentifiées à l'aide de MFA, que la demande soit réellement authentifiée ou non. Elle autorise toutes les demandes effectuées à l'aide d'informations d'identification temporaires et refuse l'accès aux informations d'identification à long terme. Utilisez cet exemple avec précaution, car il ne teste pas si l'authentification MFA a été effectivement utilisée.

AWS : EC2 Vpc InstanceSource

Cette clé identifie le VPC auquel les informations d'identification du rôle IAM Amazon EC2 ont été fournies. Vous pouvez utiliser cette clé dans une politique avec la clé `aws:SourceVPC` globale pour vérifier si un appel provient d'un VPC (`aws:SourceVPC`) qui correspond au VPC auquel un identifiant a été délivré (`aws:Ec2InstanceSourceVpc`).

- Disponibilité : cette clé est incluse dans le contexte de la demande lorsque le demandeur signe des demandes avec des informations d'identification de rôle Amazon EC2. Elle peut être utilisée dans les politiques IAM, les politiques de contrôle des services, les politiques de points de terminaison d'un VPC et les politiques de ressources.
- Type de données : chaîne
- Type de valeur – À valeur unique

Cette clé peut être utilisée avec des valeurs d'identifiant VPC, mais elle est particulièrement utile lorsqu'elle est utilisée en tant que variable associée à la clé contextuelle `aws:SourceVpc`. Cette clé contextuelle `aws:SourceVpc` figure dans le contexte de la demande uniquement si le demandeur utilise un point de terminaison de VPC pour effectuer la demande. Utiliser `aws:Ec2InstanceSourceVpc` avec `aws:SourceVpc` permet d'utiliser `aws:Ec2InstanceSourceVpc` plus largement, car la comparaison porte sur des valeurs qui changent généralement ensemble.

> **Note**
>
> Obligatoire ne figure pas dans EC2-Classic.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "RequireSameVPC",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
            "aws:SourceVpc": "${aws:Ec2InstanceSourceVpc}"
        },
        "Null": {
          "ec2:SourceInstanceARN": "false"
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

Dans l'exemple ci-dessus, l'accès est refusé si la valeur `aws:SourceVpc` n'est pas égale à la valeur `aws:Ec2InstanceSourceVpc`. La déclaration de politique est limitée aux seuls rôles utilisés en tant que rôles d'instance Amazon EC2 en testant l'existence de la clé conditionnelle `ec2:SourceInstanceARN`.

La politique permet `aws:ViaAWSService` d' AWS autoriser les demandes lorsque celles-ci sont effectuées au nom de vos rôles d'instance Amazon EC2. Par exemple, lorsque vous envoyez une demande depuis une instance Amazon EC2 à un compartiment Amazon S3 chiffré, Amazon S3 passe un appel en votre AWS KMS nom. Certaines clés ne sont pas présentes lorsque la demande est faite à AWS KMS.

AWS : EC2 PrivateIPv4 InstanceSource

Cette clé identifie l'adresse IPv4 privée de l'elastic network interface principale à laquelle les informations d'identification du rôle IAM Amazon EC2 ont été fournies. Vous devez utiliser cette clé de condition avec sa clé `aws:Ec2InstanceSourceVpc` associée pour vous assurer de disposer d'une combinaison unique d'ID VPC et d'adresse IP privée source. Utilisez cette clé avec `aws:Ec2InstanceSourceVpc` pour vous assurer qu'une demande a été faite à partir de la même adresse IP privée à laquelle les informations d'identification ont été transmises.

- Disponibilité : cette clé est incluse dans le contexte de la demande lorsque le demandeur signe des demandes avec des informations d'identification de rôle Amazon EC2. Elle peut être utilisée dans les politiques IAM, les politiques de contrôle des services, les politiques de points de terminaison d'un VPC et les politiques de ressources.
- Type de données : adresse IP
- Type de valeur – À valeur unique

> **Important**
>
> Cette clé ne doit pas être utilisée seule dans une instruction `Allow`. Les adresses IP privées ne sont par définition pas uniques d'un point de vue global. Vous devez utiliser la clé `aws:Ec2InstanceSourceVpc` chaque fois que vous utilisez la clé `aws:Ec2InstanceSourcePrivateIPv4` pour spécifier le VPC à partir duquel les informations d'identification de votre instance Amazon EC2 peuvent être utilisées.

> **Note**
>
> Obligatoire ne figure pas dans EC2-Classic.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourceVpc": "${aws:SourceVpc}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
```

```
        },
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourcePrivateIPv4": "${aws:VpcSourceIp}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
        }
    ]
}
```

lois : SourceIdentity

Utilisez cette clé pour comparer l'identité source qui a été définie par le principal à l'identité source que vous spécifiez dans la politique.

- Disponibilité — Cette clé est incluse dans le contexte de la demande une fois qu'une identité source a été définie lorsqu'un rôle est assumé à l'aide d'une commande AWS STS CLI ou d'une opération d' AWS STS `AssumeRole`API assume-role.
- Type de données : chaîne
- Type de valeur – À valeur unique

Vous pouvez utiliser cette clé dans une politique pour autoriser les acteurs ayant défini une identité source à effectuer des actions lorsqu'ils assument un rôle. AWS L'activité pour l'identité source spécifiée du rôle apparaît dans AWS CloudTrail. Cela permet aux administrateurs de déterminer plus facilement qui ou quoi a effectué des actions avec un rôle dans AWS.

Contrairement à `sts:RoleSessionName`, une fois l'identité source définie, la valeur ne peut plus être modifiée. Elle est présente dans le contexte de la demande pour toutes les actions entreprises par le rôle. La valeur persiste dans les sessions de rôle suivantes lorsque vous utilisez les

informations d'identification de session pour endosser un autre rôle. Le fait d'endosser un rôle à partir d'un autre est appelé chaînage des rôles.

La `sts:SourceIdentity`clé est présente dans la demande lorsque le principal définit initialement une identité source tout en assumant un rôle à l'aide d'une commande AWS STS CLI ou d'une opération d' AWS STS `AssumeRole`API assume-role. La clé `aws:SourceIdentity` est présente dans la demande pour toutes les actions qui sont effectuées avec une session de rôle disposant d'un ensemble d'identités source.

La politique de confiance de rôle suivante pour `CriticalRole` dans un compte `111122223333` contient une condition pour `aws:SourceIdentity` qui empêche un principal sans identité source définie sur Saanvi ou Diego d'endosser le rôle.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeRoleIfSourceIdentity",
            "Effect": "Allow",
            "Principal": {"AWS": "arn:aws:iam::123456789012:role/CriticalRole"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringLike": {
                    "aws:SourceIdentity": ["Saanvi","Diego"]
                }
            }
        }
    ]
}
```

Pour en savoir plus sur les informations relatives à l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

EC2 : RoleDelivery

Utilisez cette clé pour comparer la version du service de métadonnées d'instance figurant dans la demande signée avec les informations d'identification du rôle IAM pour Amazon EC2. Le service des métadonnées d'instance fait la distinction entre les demandes IMDSv1 et IMDSv2 pour une demande

donnée en déterminant si les en-têtes PUT ou GET, qui sont propres à IMDSv2, sont présents dans cette demande.

- Disponibilité — Cette clé est incluse dans le contexte de la demande chaque fois que la session de rôle est créée par une instance Amazon EC2.
- Type de données : numérique
- Type de valeur – À valeur unique
- Exemples de valeurs : 1,0, 2,0

Vous pouvez configurer le service des métadonnées d'instance (IMDS) sur chaque instance afin que le code local ou les utilisateurs locaux doivent utiliser IMDSv2. Lorsque vous spécifiez que IMDSv2 doit être utilisé, IMDSv1 ne fonctionne plus.

- Service de métadonnées d'instance version 1 (IMDSv1) — Une méthode de demande/réponse
- Service des métadonnées d'instance Version 2 (IMDSv2) – méthode orientée session

Pour plus d'informations sur la configuration de votre instance pour utiliser IMDSv2, consultez Configurer les options de métadonnées de l'instance.

Dans l'exemple suivant, l'accès est refusé si la RoleDelivery valeur ec2 : dans le contexte de la demande est 1.0 (IMDSv1). Cette déclaration de politique peut être appliquée de manière générale car, si la demande n'est pas signée par les informations d'identification du rôle Amazon EC2, elle n'a aucun effet.

```
{
    "Version": "2012-10-17",
    "Statement": [
                {
            "Sid": "RequireAllEc2RolesToUseV2",
            "Effect": "Deny",
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "NumericLessThan": {
                    "ec2:RoleDelivery": "2.0"
                }
            }
        }
    ]
```

```
}
```

Pour plus d'informations, consultez Exemples de politiques pour l'utilisation des métadonnées d'instance.

ec2 : Arne SourceInstance

Utilisez cette clé pour comparer l'ARN de l'instance à partir de laquelle la session du rôle a été générée.

- Disponibilité — Cette clé est incluse dans le contexte de la demande chaque fois que la session de rôle est créée par une instance Amazon EC2.
- Type de données : ARN
- Type de valeur – À valeur unique
- Exemple de valeur — arn:aws:ec2:us-west - 2:111111111111:instance/instance-id

Pour des exemples de politique, voir Autoriser une instance spécifique à afficher les ressources d'autres AWS services.

colle : RoleAssumed Par

Le AWS Glue service définit cette clé de condition pour chaque demande d' AWS API lorsqu'il AWS Glue effectue une demande en utilisant un rôle de service pour le compte du client (pas par un poste ou un point de terminaison de développeur, mais directement par le AWS Glue service). Utilisez cette clé pour vérifier si un appel à une AWS ressource provient du AWS Glue service.

- Disponibilité — Cette clé est incluse dans le contexte de la demande lorsque AWS Glue vous effectuez une demande en utilisant un rôle de service pour le compte du client.
- Type de données : chaîne
- Type de valeur – À valeur unique
- Exemple de valeur — Cette clé est toujours définie sur`glue.amazonaws.com`.

L'exemple suivant ajoute une condition permettant au AWS Glue service d'obtenir un objet depuis un compartiment Amazon S3.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
```

```
    "Resource": "arn:aws:s3:::confidential-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:RoleAssumedBy": "glue.amazonaws.com"
        }
    }
}
```

colle : CredentialIssuing Service

Le AWS Glue service définit cette clé pour chaque demande d' AWS API à l'aide d'un rôle de service provenant d'un poste de travail ou d'un point de terminaison de développeur. Utilisez cette clé pour vérifier si un appel à une AWS ressource provient d'une AWS Glue tâche ou d'un point de terminaison du développeur.

- Disponibilité — Cette clé est incluse dans le contexte de la demande lorsque vous effectuez une AWS Glue demande provenant d'un poste de travail ou d'un point de terminaison de développeur.
- Type de données : chaîne
- Type de valeur – À valeur unique
- Exemple de valeur — Cette clé est toujours définie sur`glue.amazonaws.com`.

L'exemple suivant ajoute une condition attachée à un rôle IAM utilisé par une AWS Glue tâche. Cela garantit que certaines actions sont autorisées/refusées selon que la session de rôle est utilisée ou non pour un environnement d'exécution de AWS Glue tâches.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::confidential-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:CredentialIssuingService": "glue.amazonaws.com"
        }
    }
}
```

lambda : Arn SourceFunction

Utilisez cette clé pour identifier l'ARN de la fonction Lambda auquel les informations d'identification du rôle IAM ont été fournies. Le service Lambda définit cette clé pour chaque demande d' AWS API

provenant de l'environnement d'exécution de votre fonction. Utilisez cette clé pour vérifier si un appel à une AWS ressource provient du code d'une fonction Lambda spécifique. Lambda définit également cette clé pour certaines requêtes provenant de l'extérieur de l'environnement d'exécution, telles que l'écriture de journaux CloudWatch et l'envoi de traces à X-Ray.

- Disponibilité — Cette clé est incluse dans le contexte de la demande chaque fois que le code de fonction Lambda est invoqué.
- Type de données : ARN
- Type de valeur – À valeur unique
- Exemple de valeur — arn:aws:lambda:us-east- 1:123456789012:function : TestFunction

L'exemple suivant permet à une fonction Lambda spécifique d'`s3:PutObject`accéder au bucket spécifié.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ExampleSourceFunctionArn",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
            "Condition": {
                "ArnEquals": {
                    "lambda:SourceFunctionArn": "arn:aws:lambda:us-
east-1:123456789012:function:source_lambda"
                }
            }
        }
    ]
}
```

Pour plus d'informations, consultez la section Utilisation des informations d'identification de l'environnement d'exécution Lambda dans le Guide du AWS Lambda développeur.

ssm : bras SourceInstance

Utilisez cette clé pour identifier l'ARN de l'instance AWS Systems Manager gérée auquel les informations d'identification du rôle IAM ont été fournies. Cette clé de condition n'est pas présente

lorsque la demande provient d'une instance gérée avec un rôle IAM associé à un profil d'instance Amazon EC2.

- Disponibilité — Cette clé est incluse dans le contexte de la demande chaque fois que les informations d'identification du rôle sont fournies à une instance AWS Systems Manager gérée.
- Type de données : ARN
- Type de valeur – À valeur unique
- Exemple de valeur — arn:aws:ec2:us-west - 2:111111111111:instance/instance-id

boutique d'identité : UserId

Utilisez cette clé pour comparer l'identité du personnel d'IAM Identity Center figurant dans la demande signée avec l'identité spécifiée dans la politique.

- Disponibilité : cette clé est incluse lorsque l'appelant de la demande est un utilisateur d'IAM Identity Center.
- Type de données : chaîne
- Type de valeur – À valeur unique
- Exemple de valeur : 94482488-3041-7026-18f3-be45837cd0e4

Vous pouvez trouver le nom UserId d'un utilisateur dans IAM Identity Center en adressant une demande à l'API GetUserId à l'aide de l' AWS CLI AWS API ou du AWS SDK.

Propriétés du réseau

Utilisez les clés de condition suivantes pour comparer les détails du réseau d'où provient ou est passée la demande avec les propriétés du réseau que vous spécifiez dans la politique.

Table des matières

- lois : SourceIp
- lois : SourceVpc
- lois : SourceVpce
- lois : VpcSource IP

lois : SourceIp

Utilisez cette clé pour comparer l'adresse IP du demandeur avec celle spécifiée dans la politique. La clé de condition `aws:SourceIp` ne peut être utilisée que pour les plages d'adresses IP publiques.

- Availability (Disponibilité) : cette clé figure dans le contexte de la demande, sauf lorsque le demandeur utilise un point de terminaison VPC pour effectuer la demande.
- Type de données : adresse IP
- Type de valeur – À valeur unique

La clé de condition `aws:SourceIp` peut être utilisée dans une politique pour autoriser des principaux à effectuer des demandes uniquement dans une plage IP spécifiée.

> **Note**
>
> `aws:SourceIp` prend en charge des adresses ou une plage d'adresses IP IPv4 et IPv6. Pour obtenir la liste des Services AWS solutions compatibles IPv6, consultez le guide de l'utilisateur Amazon VPC Services AWS qui prend en charge l'IPv6.

Par exemple, vous pouvez attacher la politique basée sur l'identité suivante à un rôle IAM. Cette politique permet à l'utilisateur de placer des objets dans le compartiment Amazon S3 `DOC-EXAMPLE-BUCKET3` si l'appel est effectué à partir de la plage d'adresses IPv4 spécifiée. Cette politique permet également à un AWS service utilisé Transmission des sessions d'accès d'effectuer cette opération en votre nom.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET3/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
```

```
        }
    ]
}
```

Si vous devez restreindre l'accès depuis les réseaux qui prennent en charge à la fois l'adressage IPv4 et IPv6, vous pouvez inclure l'adresse ou les plages d'adresses IPv4 et IPv6 dans la condition de politique IAM. La politique basée sur l'identité suivante permet à l'utilisateur de placer des objets dans le compartiment Amazon S3 `DOC-EXAMPLE-BUCKET3` si l'utilisateur effectue l'appel à partir des plages d'adresses IPv4 ou IPv6 spécifiées. Avant d'inclure des plages d'adresses IPv6 dans votre politique IAM, vérifiez que celles avec lesquelles Service AWS vous travaillez prennent en charge le protocole IPv6. Pour obtenir la liste Services AWS des solutions compatibles IPv6, consultez Services AWS le manuel Amazon VPC User Guide.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET3/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": [
                        "203.0.113.0/24",
                        "2001:DB8:1234:5678::/64"
                    ]
                }
            }
        }
    ]
}
```

Si la demande provient d'un hôte qui utilise un point de terminaison Amazon VPC, la clé `aws:SourceIp` n'est pas disponible. Vous devez plutôt utiliser une clé spécifique au VPC, telle que aws : VpcSource Ip. Pour plus d'informations sur l'utilisation des points de terminaison d'un VPC, veuillez consulter la rubrique Gestion des identités et des accès pour les points de terminaison d'un VPC et les services de points de terminaison d'un VPC dans le Guide AWS PrivateLink .

lois : SourceVpc

Utilisez cette clé pour vérifier si la demande transite par le VPC auquel le point de terminaison du VPC est attaché. Dans une politique, vous pouvez utiliser cette clé pour autoriser l'accès à un seul VPC spécifique. Pour de plus amples informations, veuillez consulter Restriction de l'accès à un VPC spécifique dans le Guide de l'utilisateur service de stockage simple Amazon.

- Availability (Disponibilité) : cette clé figure dans le contexte de la demande uniquement si le demandeur utilise un point de terminaison de VPC pour effectuer la demande.
- Type de données : chaîne
- Type de valeur – À valeur unique

lois : SourceVpce

Utilisez cette clé pour comparer l'identifiant du point de terminaison de VPC de la demande avec l'ID du point de terminaison spécifié dans la politique. Dans une politique, vous pouvez utiliser cette clé pour restreindre l'accès à un point de terminaison de VPC spécifique. Pour plus d'informations, veuillez consulter Restriction de l'accès à un point de terminaison d'un VPC spécifique dans le Guide de l'utilisateur service de stockage simple Amazon.

- Availability (Disponibilité) : cette clé figure dans le contexte de la demande uniquement si le demandeur utilise un point de terminaison de VPC pour effectuer la demande.
- Type de données : chaîne
- Type de valeur – À valeur unique

lois : VpcSource IP

Utilisez cette clé pour comparer l'adresse IP à partir de laquelle une demande a été effectuée avec celle spécifiée dans la politique. Dans une politique, la clé ne correspond que si la demande provient de l'adresse IP spécifiée et qu'elle passe par un point de terminaison de VPC.

- Availability (Disponibilité) : cette clé ne figure dans le contexte de la demande que si la demande est effectuée à l'aide d'un point de terminaison de VPC.
- Type de données : adresse IP
- Type de valeur – À valeur unique

Pour en savoir plus, consultez Contrôle de l’accès aux services avec des points de terminaison d’un VPC dans le guide de l’utilisateur Amazon VPC.

> **Note**
>
> `aws:VpcSourceIp` prend en charge des adresses ou une plage d'adresses IP IPv4 et IPv6. Pour obtenir la liste Services AWS des solutions compatibles IPv6, consultez Services AWS le manuel Amazon VPC User Guide.

Propriétés de la ressource

Utilisez les clés de condition suivantes pour comparer les détails de la ressource cible de la demande avec les propriétés de ressource que vous spécifiez dans la politique.

Table des matières

- lois : ResourceAccount
- aws : ResourceOrg Chemins
- lois : ResourceOrg ID
- aws :ResourceTag//tag-key

lois : ResourceAccount

Utilisez cette clé pour comparer l'ID de l'Compte AWS du propriétaire de la ressource demandée avec le compte de la ressource dans la politique. Vous pouvez ensuite autoriser ou refuser l'accès à cette ressource en fonction du compte propriétaire de la ressource.

- Disponibilité : cette clé figure toujours dans le contexte de la demande pour la plupart des services. Les actions suivantes ne prennent pas en charge cette clé :
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - Amazon Elastic Block Store : toutes les actions
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`

- `ec2:AcceptVpcEndpointConnections`
- `ec2:AcceptVpcPeeringConnection`
- `ec2:CopyImage`
- `ec2:CopySnapshot`
- `ec2:CreateTransitGatewayPeeringAttachment`
- `ec2:CreateVolume`
- `ec2:CreateVpcEndpoint`
- `ec2:CreateVpcPeeringConnection`
- `ec2:DeleteTransitGatewayPeeringAttachment`
- `ec2:DeleteVpcPeeringConnection`
- `ec2:RejectTransitGatewayPeeringAttachment`
- `ec2:RejectVpcEndpointConnections`
- `ec2:RejectVpcPeeringConnection`

- Amazon EventBridge
  - `events:PutEvents`— EventBridge `PutEvents` appels sur un bus d'événements dans un autre compte, si ce bus d'événements a été configuré comme EventBridge cible multi-comptes avant le 2 mars 2023. Pour plus d'informations, consultez la section Accorder des autorisations pour autoriser des événements provenant d'autres AWS comptes dans le guide de EventBridge l'utilisateur Amazon.
- Amazon GuardDuty
  - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
  - `macie2:AcceptInvitation`
- Amazon OpenSearch Service
  - `es:AcceptInboundConnection`
  - `es:CreateOutboundConnection`
- Amazon Route 53
  - `route53:AssociateVpcWithHostedZone`
  - `route53:CreateVPCAssociationAuthorization`
  - `route53:DeleteVPCAssociationAuthorization`
  - `route53:DisassociateVPCFromHostedZone`

    - `route53:ListHostedZonesByVPC`
  - AWS Security Hub
    - `securityhub:AcceptAdministratorInvitation`
- Type de données : chaîne
- Type de valeur – À valeur unique

> **Note**
>
> Pour des considérations supplémentaires concernant les actions non prises en charge ci-dessus, consultez le référentiel Exemples de politiques relatives aux périmètres de données (français non garanti).

Cette clé est égale à l' Compte AWS identifiant du compte dont les ressources ont été évaluées dans la demande.

Pour la plupart des ressources de votre compte, l'ARN contient l'ID du compte propriétaire de cette ressource. Pour certaines ressources, telles que les compartiments Amazon S3, l'ARN de ressource n'inclut pas l'ID de compte. Les deux exemples suivants montrent la différence entre une ressource avec un ID de compte dans l'ARN et un ARN Amazon S3 sans ID de compte :

- `arn:aws:iam::123456789012:role/AWSExampleRole` – rôle IAM créé et détenu dans le compte 123456789012.
- `arn:aws:s3:::DOC-EXAMPLE-BUCKET2` : compartiment Amazon S3 créé et détenu dans le compte `111122223333`, non affiché dans l'ARN.

Utilisez la AWS console, ou l'API, ou la CLI, pour trouver toutes vos ressources et les ARN correspondants.

Vous élaborez une politique qui refuse les autorisations d'accès aux ressources en fonction de l'ID de compte du propriétaire de la ressource. Par exemple, la politique basée sur l'identité suivante refuse l'accès à la ressource spécifiée si celle-ci n'appartient pas au compte spécifié.

Pour utiliser cette politique, remplacez le texte en italique de l'espace réservé dans l'exemple de politique par vos propres informations de compte.

Important

Cette politique ne permet aucune action. Au lieu de cela, elle utilise l'effet `Deny` qui refuse explicitement l'accès à toutes les ressources répertoriées dans l'instruction n'appartenant pas au compte répertorié. Utilisez cette politique en combinaison avec d'autres politiques qui autorisent l'accès à des ressources spécifiques.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyInteractionWithResourcesNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:region:account:*"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "account"
          ]
        }
      }
    }
  ]
}
```

Cette politique refuse l'accès à toutes les ressources pour un AWS service spécifique, sauf si le fournisseur spécifié Compte AWS est propriétaire de la ressource.

Note

Certains Services AWS nécessitent l'accès à des ressources AWS détenues qui sont hébergées dans un autre Compte AWS. L'utilisation de `aws:ResourceAccount` dans vos politiques basées sur l'identité peut avoir un impact sur la capacité de votre identité à accéder à ces ressources.

Certains AWS services, tels que AWS Data Exchange, dépendent de l'accès à des ressources extérieures aux vôtres Comptes AWS pour leurs opérations normales. Si vous utilisez l'élément `aws:ResourceAccount` dans vos politiques, incluez des déclarations supplémentaires afin de créer des dérogations pour ces services. L'exemple de politique AWS: Refusez l'accès aux ressources Amazon S3 en dehors de votre compte, sauf AWS Data Exchange montre comment refuser l'accès en fonction du compte de la ressource tout en définissant des exceptions pour les ressources appartenant au service.

Utilisez cet exemple de politique comme modèle pour créer vos propres politiques personnalisées. Reportez-vous à votre documentation de service pour plus d'informations.

aws : ResourceOrg Chemins

Utilisez cette clé pour comparer le chemin AWS des Organizations pour la ressource consultée avec le chemin indiqué dans la politique. Dans une politique, cette clé de condition garantit que la ressource appartient à un membre du compte appartenant à la racine de l'organisation ou aux unités organisationnelles (UO) spécifiées dans AWS Organizations. Un chemin AWS Organizations est une représentation textuelle de la structure d'une entité Organizations. Pour de plus amples informations sur l'utilisation et la compréhension des chemins, veuillez consulter Comprendre le chemin d'entité AWS Organizations.

- Disponibilité : cette clé ne figure dans le contexte de la demande que si le compte qui possède la ressource est membre d'une organisation. Cette clé de condition globale ne prend pas en charge les actions suivantes :
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - Amazon Elastic Block Store : toutes les actions
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CopyImage`
    - `ec2:CopySnapshot`
    - `ec2:CreateTransitGatewayPeeringAttachment`

    - `ec2:CreateVolume`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
    - `ec2:DeleteTransitGatewayPeeringAttachment`
    - `ec2:DeleteVpcPeeringConnection`
    - `ec2:RejectTransitGatewayPeeringAttachment`
    - `ec2:RejectVpcEndpointConnections`
    - `ec2:RejectVpcPeeringConnection`
- Amazon EventBridge
    - `events:PutEvents`— EventBridge `PutEvents` appels sur un bus d'événements dans un autre compte, si ce bus d'événements a été configuré comme EventBridge cible multi-comptes avant le 2 mars 2023. Pour plus d'informations, consultez la section Accorder des autorisations pour autoriser des événements provenant d'autres AWS comptes dans le guide de EventBridge l'utilisateur Amazon.
- Amazon GuardDuty
    - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
    - `macie2:AcceptInvitation`
- Amazon OpenSearch Service
    - `es:AcceptInboundConnection`
    - `es:CreateOutboundConnection`
- Amazon Route 53
    - `route53:AssociateVpcWithHostedZone`
    - `route53:CreateVPCAssociationAuthorization`
    - `route53:DeleteVPCAssociationAuthorization`
    - `route53:DisassociateVPCFromHostedZone`
    - `route53:ListHostedZonesByVPC`
- AWS Security Hub
    - `securityhub:AcceptAdministratorInvitation`

- Type de données — Chaîne (liste)
- Type de valeur – À valeur multiple

> **Note**
>
> Pour des considérations supplémentaires concernant les actions non prises en charge ci-dessus, consultez le référentiel Exemples de politiques relatives aux périmètres de données (français non garanti).

`aws:ResourceOrgPaths` est une clé de condition à valeurs multiples. Les clés multivaluées peuvent avoir plusieurs valeurs dans le contexte de la demande. Vous devez utiliser les opérateurs d'ensemble `ForAnyValue` ou `ForAllValues` avec l'opérateur de condition de chaîne pour cette clé. Pour de plus amples informations sur les clés de condition à valeurs multiples, veuillez consulter Clés de contexte à valeurs multiples.

Par exemple, la condition suivante renvoie `True` pour les ressources qui appartiennent à l'organisation `o-a1b2c3d4e5`. Lorsque vous incluez un caractère générique, vous devez utiliser l'opérateur de StringLikecondition.

```
"Condition": {
      "ForAnyValue:StringLike": {
            "aws:ResourceOrgPaths":["o-a1b2c3d4e5/*"]
   }
}
```

La condition suivante renvoie `True` pour les ressources avec l'ID d'unité d'organisation `ou-ab12-11111111`. Elle correspond aux ressources détenues par les comptes rattachés à l'unité d'organisation ou-ab12-11111111 ou à l'une des unités d'organisation enfants.

```
"Condition": { "ForAnyValue:StringLike" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/*"]
}}
```

La condition suivante renvoie `True` pour les ressources détenues par des comptes rattachés directement à l'ID d'unité d'organisation `ou-ab12-22222222`, mais pas aux unités d'organisation enfants. L'exemple suivant utilise l'opérateur de StringEqualscondition pour spécifier l'exigence de correspondance exacte pour l'ID d'unité d'organisation et non une correspondance générique.

```
"Condition": { "ForAnyValue:StringEquals" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

> **Note**
>
> Certains Services AWS nécessitent l'accès à des ressources AWS détenues qui sont hébergées dans un autre Compte AWS. L'utilisation de `aws:ResourceOrgPaths` dans vos politiques basées sur l'identité peut avoir un impact sur la capacité de votre identité à accéder à ces ressources.

Certains AWS services, tels que AWS Data Exchange, dépendent de l'accès à des ressources extérieures aux vôtres Comptes AWS pour leurs opérations normales. Si vous utilisez la clé `aws:ResourceOrgPaths` dans vos politiques, incluez des déclarations supplémentaires afin de créer des dérogations pour ces services. L'exemple de politique AWS: Refusez l'accès aux ressources Amazon S3 en dehors de votre compte, sauf AWS Data Exchange montre comment refuser l'accès en fonction du compte de la ressource tout en définissant des exceptions pour les ressources appartenant au service. Vous pouvez créer une politique similaire pour restreindre l'accès aux ressources d'une unité d'organisation (UO) à l'aide de la clé `aws:ResourceOrgPaths`, tout en tenant compte des ressources appartenant au service.

Utilisez cet exemple de politique comme modèle pour créer vos propres politiques personnalisées. Reportez-vous à votre documentation de service pour plus d'informations.

lois : ResourceOrg ID

Utilisez cette clé pour comparer l'identifiant de l'organisation dans AWS Organizations à laquelle appartient la ressource demandée avec l'identifiant spécifié dans la politique.

- Disponibilité : cette clé ne figure dans le contexte de la demande que si le compte qui possède la ressource est membre d'une organisation. Cette clé de condition globale ne prend pas en charge les actions suivantes :
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Detective
    - `detective:AcceptInvitation`
  - Amazon Elastic Block Store : toutes les actions
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`

  - `ec2:AcceptVpcPeeringConnection`
  - `ec2:CopyImage`
  - `ec2:CopySnapshot`
  - `ec2:CreateTransitGatewayPeeringAttachment`
  - `ec2:CreateVolume`
  - `ec2:CreateVpcEndpoint`
  - `ec2:CreateVpcPeeringConnection`
  - `ec2:DeleteTransitGatewayPeeringAttachment`
  - `ec2:DeleteVpcPeeringConnection`
  - `ec2:RejectTransitGatewayPeeringAttachment`
  - `ec2:RejectVpcEndpointConnections`
  - `ec2:RejectVpcPeeringConnection`
- Amazon EventBridge
  - `events:PutEvents`— EventBridge `PutEvents` appels sur un bus d'événements dans un autre compte, si ce bus d'événements a été configuré comme EventBridge cible multi-comptes avant le 2 mars 2023. Pour plus d'informations, consultez la section Accorder des autorisations pour autoriser des événements provenant d'autres AWS comptes dans le guide de EventBridge l'utilisateur Amazon.
- Amazon GuardDuty
  - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
  - `macie2:AcceptInvitation`
- Amazon OpenSearch Service
  - `es:AcceptInboundConnection`
  - `es:CreateOutboundConnection`
- Amazon Route 53
  - `route53:AssociateVpcWithHostedZone`
  - `route53:CreateVPCAssociationAuthorization`
  - `route53:DeleteVPCAssociationAuthorization`
  - `route53:DisassociateVPCFromHostedZone`
  - `route53:ListHostedZonesByVPC`

  - AWS Security Hub
    - `securityhub:AcceptAdministratorInvitation`
- Type de données : chaîne
- Type de valeur – À valeur unique

> **Note**
>
> Pour des considérations supplémentaires concernant les actions non prises en charge ci-dessus, consultez le référentiel Exemples de politiques relatives aux périmètres de données (français non garanti).

Cette clé globale renvoie l'ID de l'organisation de la ressource pour une demande donnée. Elle vous permet de créer des règles qui s'appliquent à toutes les ressources d'une organisation qui sont spécifiées dans l'élément `Resource` d'une politique basée sur l'identité. Vous pouvez spécifier l'ID de l'organisation dans l'élément Condition. Lorsque vous ajoutez et supprimez des comptes, les politiques qui contiennent la clé `aws:ResourceOrgID` incluent automatiquement les bons comptes et vous n'avez pas besoin de la mettre à jour manuellement.

Par exemple, la politique suivante empêche le principal d'ajouter des objets à la ressource `policy-genius-dev`, sauf si la ressource Amazon S3 appartient à la même organisation que le principal qui effectue la demande.

> **Important**
>
> Cette politique ne permet aucune action. Au lieu de cela, elle utilise l'effet `Deny` qui refuse explicitement l'accès à toutes les ressources répertoriées dans l'instruction n'appartenant pas au compte répertorié. Utilisez cette politique en combinaison avec d'autres politiques qui autorisent l'accès à des ressources spécifiques.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "DenyPutObjectToS3ResourcesOutsideMyOrganization",
    "Effect": "Deny",
    "Action": "s3:PutObject",
```

```
    "Resource": "arn:partition:s3:::policy-genius-dev/*",
    "Condition": {
      "StringNotEquals": {
        "aws:ResourceOrgID": "${aws:PrincipalOrgID}"
      }
    }
  }
}
```

> **Note**
>
> Certains Services AWS nécessitent l'accès à des ressources AWS détenues qui sont hébergées dans un autre Compte AWS. L'utilisation de `aws:ResourceOrgID` dans vos politiques basées sur l'identité peut avoir un impact sur la capacité de votre identité à accéder à ces ressources.

Certains AWS services, tels que AWS Data Exchange, dépendent de l'accès à des ressources extérieures aux vôtres Comptes AWS pour leurs opérations normales. Si vous utilisez la clé `aws:ResourceOrgID` dans vos politiques, incluez des déclarations supplémentaires afin de créer des dérogations pour ces services. L'exemple de politique AWS: Refusez l'accès aux ressources Amazon S3 en dehors de votre compte, sauf AWS Data Exchange montre comment refuser l'accès en fonction du compte de la ressource tout en définissant des exceptions pour les ressources appartenant au service. Vous pouvez créer une politique similaire pour restreindre l'accès aux ressources de votre organisation à l'aide de la clé `aws:ResourceOrgID`, tout en tenant compte des ressources appartenant au service.

Utilisez cet exemple de politique comme modèle pour créer vos propres politiques personnalisées. Reportez-vous à votre documentation de service pour plus d'informations.

Dans la vidéo suivante, découvrez comment utiliser la clé de condition `aws:ResourceOrgID` dans une politique.

Assurez-vous que les identités et les réseaux ne peuvent être utilisés que pour accéder à des ressources fiables.

aws :ResourceTag//tag-key

Utilisez cette clé pour comparer la paire valeur clé d'étiquette que vous spécifiez dans la politique avec la paire valeur clé attachée à la ressource. Par exemple, vous pouvez exiger que l'accès à une

ressource soit autorisé uniquement si la clé de balise `"Dept"` est attachée à la ressource avec la valeur `"Marketing"`. Pour plus d'informations, consultez Contrôle de l'accès aux ressources AWS.

- Disponibilité : cette clé figure dans le contexte de la demande lorsque la ressource demandée possède déjà des balises attachées ou dans les demandes qui créent une ressource avec une balise attachée. Cette clé est renvoyée uniquement pour les ressources qui prennent en charge l'autorisation basée sur les balises. Il y a une clé de contexte pour chaque paire clé-valeur de balise.
- Type de données : chaîne
- Type de valeur – À valeur unique

Cette clé de contexte est formatée selon le schéma `"aws:ResourceTag/`*`tag-key`*`":"`*`tag-value`*`"`, où *`tag-key`* et *`tag-value`* représentent respectivement une clé de balise et une paire de valeurs. Les clés et les valeurs d'étiquette ne sont pas sensibles à la casse. Cela signifie que si vous spécifiez `"aws:ResourceTag/TagKey1": "Value1"` dans l'élément de condition de votre politique, la condition correspond à une clé de balise de ressource nommée `TagKey1` ou `tagkey1`, mais pas aux deux.

Pour obtenir des exemples d'utilisation de la clé `aws:ResourceTag` pour contrôler l'accès aux ressources IAM, consultez la section Contrôle de l'accès aux ressources AWS.

Pour des exemples d'utilisation de la `aws:ResourceTag` clé pour contrôler l'accès à d'autres AWS ressources, consultezContrôle de l'accès aux AWS ressources à l'aide de balises.

Pour obtenir un didacticiel sur l'utilisation de la clé de condition `aws:ResourceTag` pour le contrôle d'accès basé sur les attributs (ABAC), reportez-vous à la section Tutoriel IAM : définir les autorisations d'accès aux AWS ressources en fonction des balises.

Propriétés de la demande

Utilisez les clés de condition suivantes pour comparer les détails de la demande elle-même et son contenu avec les propriétés de demande que vous spécifiez dans la politique.

Table des matières

- lois : CalledVia
- aws : CalledVia Tout d'abord
- aws : CalledVia Dernier

- AWS : via AWSService
- lois : CurrentTime
- lois : EpochTime
- aws:referer
- lois : RequestedRegion
- aws :RequestTag//tag-key
- lois : TagKeys
- lois : SecureTransport
- lois : SourceArn
- lois : SourceAccount
- aws : SourceOrg Chemins
- lois : SourceOrg ID
- lois : UserAgent

lois : CalledVia

Utilisez cette clé pour comparer les services de la politique avec les services qui ont créé des demandes au nom du principal IAM (utilisateur ou rôle). Lorsqu'un principal adresse une demande à un AWS service, ce service peut utiliser les informations d'identification du principal pour faire des demandes ultérieures à d'autres services. La clé `aws:CalledVia` contient une liste ordonnée des services de la chaîne ayant effectué des demandes pour le compte du principal.

Par exemple, vous pouvez l'utiliser AWS CloudFormation pour lire et écrire à partir d'une table Amazon DynamoDB. DynamoDB utilise ensuite le chiffrement fourni par AWS Key Management Service ().AWS KMS

- Availability (Disponibilité): cette clé est présente dans la demande lorsqu'un service prenant en charge `aws:CalledVia` utilise les informations d'identification d'un principal IAM pour effectuer une demande à un autre service. Cette clé n'est pas présente si le service utilise un rôle de service ou un rôle lié à un service pour effectuer un appel au nom du mandataire. Cette clé n'est pas non plus présente lorsque le principal effectue directement l'appel.
- Type de données — Chaîne (liste)
- Type de valeur – À valeur multiple

Pour utiliser la clé de `aws:CalledVia` condition dans une politique, vous devez fournir les principaux de service permettant d'autoriser ou de refuser les demandes AWS de service. AWS prend en charge l'utilisation des principes de service suivants avec`aws:CalledVia`.

Principal du service
aoss.amazonaws.com
athena.amazonaws.com
backup.amazonaws.com
cloud9.amazonaws.com
cloudformation.amazonaws.com
databrew.amazonaws.com
dataexchange.amazonaws.com
dynamodb.amazonaws.com
imagebuilder.amazonaws.com
kms.amazonaws.com
mgn.amazonaws.com
nimble.amazonaws.com
omics.amazonaws.com
ram.amazonaws.com
robomaker.amazonaws.com
servicecatalog-appregistry.amazonaws.com
sqlworkbench.amazonaws.com
ssm-guiconnect.amazonaws.com

Pour autoriser ou refuser l'accès lorsqu'un service effectue une demande à l'aide des informations d'identification du principal, utilisez la clé de condition `AWS : via AWSService`. Cette clé de condition soutient les AWS services.

La clé `aws:CalledVia` est une clé à valeurs multiples. Cependant, vous ne pouvez pas imposer l'ordre en utilisant cette clé dans une condition. Dans l'exemple ci-dessus, User 1 (Utilisateur 1) effectue une demande à AWS CloudFormation, qui appelle DynamoDB, qui appelle AWS KMS. Il s'agit de trois demandes distinctes. Le dernier appel à AWS KMS est effectué par l'utilisateur 1 via DynamoDB, AWS CloudFormation puis par DynamoDB.

Dans ce cas, la clé `aws:CalledVia` dans le contexte de la demande comprend `cloudformation.amazonaws.com` et `dynamodb.amazonaws.com`, dans cet ordre. Si vous vous souciez uniquement du fait que l'appel a été effectué via DynamoDB quelque part dans la chaîne des demandes, vous pouvez utiliser cette clé de condition dans votre politique.

Par exemple, la politique suivante permet de gérer la AWS KMS clé nomméemy-example-key, mais uniquement si DynamoDB est l'un des services demandeurs. L'opérateur de condition `ForAnyValue:StringEquals` s'assure que DynamoDB est un des services appelant. Si le principal effectue directement l'appel à AWS KMS , la condition renvoie `false` et la demande n'est pas autorisée par cette politique.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaDynamodb",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:region:111122223333:key/my-example-key",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": ["dynamodb.amazonaws.com"]
                }
            }
        }
```

```
    ]
}
```

Si vous souhaitez définir le service effectuant le premier ou le dernier appel de la chaîne, vous pouvez utiliser les clés aws:CalledViaLast et aws:CalledViaFirst. Par exemple, la politique suivante permet de gérer la clé nommée my-example-key dans AWS KMS. Ces AWS KMS opérations ne sont autorisées que si plusieurs demandes ont été incluses dans la chaîne. La première demande doit être faite via AWS CloudFormation et la dernière, via DynamoDB. Si d'autres services font des demandes au milieu de la chaîne, l'opération est toujours autorisée.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaChain",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:region:111122223333:key/my-example-key",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaFirst": "cloudformation.amazonaws.com",
                    "aws:CalledViaLast": "dynamodb.amazonaws.com"
                }
            }
        }
    ]
}
```

Les clés aws:CalledViaFirst et aws:CalledViaLast sont présentes dans la demande lorsqu'un service utilise les informations d'identification d'un principal IAM pour appeler un autre service. Elles indiquent le premier et le dernier services ayant effectué des appels dans la chaîne de demandes. Supposons, par exemple, qu' AWS CloudFormation un autre service appelé DynamoDB appelle DynamoDBX Service, qui appelle ensuite. AWS KMS Le dernier appel à AWS KMS est effectué par User 1 via AWS CloudFormation, thenX Service, puis DynamoDB. Il a d'abord été appelé via AWS CloudFormation et appelé pour la dernière fois via DynamoDB.

aws : CalledVia Tout d'abord

Utilisez cette clé pour comparer les services de la politique avec le premier service ayant fait une demande au nom du principal IAM (utilisateur ou rôle). Pour de plus amples informations, veuillez consulter `aws:CalledVia`.

- Availability (Disponibilité) : cette clé est présente dans la requête lorsqu'un service utilise les informations d'identification d'un principal IAM pour effectuer au moins une autre requête à un service différent. Cette clé n'est pas présente si le service utilise un rôle de service ou un rôle lié à un service pour effectuer un appel au nom du mandataire. Cette clé n'est pas non plus présente lorsque le principal effectue directement l'appel.
- Type de données : chaîne
- Type de valeur – À valeur unique

aws : CalledVia Dernier

Utilisez cette clé pour comparer les services de la politique avec le dernier service ayant fait une demande au nom du principal IAM (utilisateur ou rôle). Pour plus d'informations, veuillez consulter `aws:CalledVia`.

- Availability (Disponibilité) : cette clé est présente dans la requête lorsqu'un service utilise les informations d'identification d'un principal IAM pour effectuer au moins une autre requête à un service différent. Cette clé n'est pas présente si le service utilise un rôle de service ou un rôle lié à un service pour effectuer un appel au nom du mandataire. Cette clé n'est pas non plus présente lorsque le principal effectue directement l'appel.
- Type de données : chaîne
- Type de valeur – À valeur unique

AWS : via AWSService

Utilisez cette clé pour vérifier si un AWS service fait une demande à un autre service en votre nom.

La clé de contexte de demande renvoie `true` lorsqu'un service utilise les informations d'identification d'un principal IAM pour effectuer une demande au nom du principal. La clé de contexte renvoie `false` si le service utilise un rôle de service ou un rôle lié à un service pour effectuer un appel au nom du principal. La clé de contexte de demande renvoie également `false` lorsque le principal effectue l'appel directement.

- Availability (Disponibilité) : cette clé figure toujours dans le contexte de la demande.
- Type de données : booléen
- Type de valeur – À valeur unique

Vous pouvez utiliser cette clé de condition pour autoriser ou refuser l'accès selon qu'une demande a été effectuée par un service.

lois : CurrentTime

Utilisez cette clé pour comparer la date et l'heure de la demande avec celles spécifiée dans la politique. Pour visualiser un exemple de politique qui utilise cette clé de condition, veuillez consulter AWS : permet l'accès en fonction de la date et de l'heure.

- Availability (Disponibilité) : cette clé figure toujours dans le contexte de la demande.
- Type de données — Date
- Type de valeur – À valeur unique

lois : EpochTime

Utilisez cette clé pour comparer la date et l'heure de la demande au format Epoch ou Unix avec la valeur spécifiée dans la politique. Cette clé accepte également le nombre de secondes depuis le 1er janvier 1970.

- Availability (Disponibilité) : cette clé figure toujours dans le contexte de la demande.
- Type de données : date, numérique
- Type de valeur – À valeur unique

aws:referer

Utilisez cette clé pour comparer le référent de la demande dans le navigateur client avec le référent spécifié dans la politique. La valeur `aws:referer` du contexte de la demande est fournie par le principal dans un en-tête HTTP. L'en-tête `Referer` est inclus dans une demande de navigateur Web lorsque vous sélectionnez un lien sur une page Web. L'en-tête `Referer` contient l'URL de la page Web où le lien a été sélectionné.

- Disponibilité — Cette clé est incluse dans le contexte de la demande uniquement si la demande à la AWS ressource a été invoquée par un lien depuis l'URL d'une page Web dans le navigateur.

Cette clé n'est pas incluse pour les demandes de programmation, car elle n'utilise pas de lien de navigateur pour accéder à la ressource AWS .

- Type de données : chaîne
- Type de valeur – À valeur unique

Par exemple, vous pouvez accéder à un objet Amazon S3 directement à l'aide d'une URL ou en utilisant l'invocation directe de l'API. Pour plus d'informations, consultez la section Opérations d'API Amazon S3 directement à l'aide d'un navigateur web. Lorsque vous accédez à un objet Amazon S3 à partir d'une URL qui existe dans une page web, l'URL de la page web source est utilisée dans `aws:referer`. Lorsque vous accédez à un objet Amazon S3 en saisissant l'URL dans votre navigateur, `aws:referer` n'est pas présent. Lorsque vous appelez l'API directement, `aws:referer` n'est pas non plus présent. Vous pouvez utiliser la clé de condition `aws:referer` dans une politique pour autoriser les demandes effectuées à partir d'un référent spécifique, comme un lien sur une page Web dans le domaine de votre entreprise.

> **Warning**
>
> Utilisez cette clé avec précaution. Il est dangereux d'inclure une valeur d'en-tête de référent connue publiquement. Les tiers non autorisés peuvent utiliser des navigateurs modifiés ou personnalisés pour fournir n'importe quelle valeur `aws:referer` de leur choix. Par conséquent, `aws:referer` il ne doit pas être utilisé pour empêcher des parties non autorisées de faire des AWS demandes directes. Cette clé est fournie uniquement pour permettre aux clients de protéger leur contenu numérique, stocké notamment dans Amazon S3, contre tout référencement sur des sites tiers non autorisés.

lois : RequestedRegion

Utilisez cette clé pour comparer la AWS région appelée dans la demande avec la région que vous spécifiez dans la politique. Vous pouvez utiliser cette clé de condition globale pour contrôler les régions qui peuvent être demandées. Pour consulter les AWS régions de chaque service, consultez la section Points de terminaison et quotas du service dans le Référence générale d'Amazon Web Services.

- Availability (Disponibilité) : cette clé figure toujours dans le contexte de la demande.
- Type de données : chaîne
- Type de valeur – À valeur unique

Certains services globaux, tels qu'IAM, ont un point de terminaison unique. Comme ce point de terminaison se trouve physiquement dans la région USA Est (Virginie du Nord), les appels IAM sont toujours effectués vers la région us-east-1. Par exemple, si vous créez une politique qui refuse l'accès à tous les services si la région demandée n'est pas us-west-2, les appels IAM échouent toujours. Pour voir un exemple de la manière de contourner ce problème, voir NotAction avec Deny.

> **Note**
>
> La clé de condition `aws:RequestedRegion` vous permet de contrôler le point de terminaison de service qui est appelé, mais n'a pas d'impact sur l'opération. Certains services ont des impacts entre régions.
> Par exemple, Amazon S3 possède des opérations d'API qui s'étendent entre les régions.
>
> - Vous pouvez appeler `s3:PutBucketReplication` dans une région (qui est affectée par la clé de condition `aws:RequestedRegion`), mais d'autres régions sont affectées en fonction des paramètres de configuration des réplications.
> - Vous pouvez invoquer `s3:CreateBucket` pour créer un compartiment dans une autre région et utiliser la clé de condition `s3:LocationConstraint` pour contrôler les régions applicables.

Vous pouvez utiliser cette clé de contexte pour limiter l'accès aux AWS services au sein d'un ensemble donné de régions. Par exemple, la politique suivante permet à un utilisateur d'afficher toutes les instances Amazon EC2 dans AWS Management Console. Cependant, elle lui permet d'apporter des modifications aux instances dans Irlande (eu-west-1), Londres (eu-west-2) ou Paris (eu-west-3).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "InstanceConsoleReadOnly",
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:Export*",
                "ec2:Get*",
                "ec2:Search*"
            ],
```

```
            "Resource": "*"
        },
        {
            "Sid": "InstanceWriteRegionRestricted",
            "Effect": "Allow",
            "Action": [
                "ec2:Associate*",
                "ec2:Import*",
                "ec2:Modify*",
                "ec2:Monitor*",
                "ec2:Reset*",
                "ec2:Run*",
                "ec2:Start*",
                "ec2:Stop*",
                "ec2:Terminate*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestedRegion": [
                        "eu-west-1",
                        "eu-west-2",
                        "eu-west-3"
                    ]
                }
            }
        }
    ]
}
```

aws :RequestTag//tag-key

Utilisez cette clé pour comparer la paire clé-valeur de balise qui a été transmise dans la demande avec la paire de balises spécifiée dans la politique. Par exemple, vous pouvez vérifier que la demande comprend la clé de balise `"Dept"` et qu'elle a la valeur `"Accounting"`. Pour plus d'informations, consultez Contrôle de l'accès au cours des demandes AWS.

- Disponibilité : cette clé figure dans le contexte de la demande lorsque les balises sont transmises dans la demande. Lorsque plusieurs balises sont transmises dans la demande, il y a une clé de contexte pour chaque paire clé-valeur de balise.
- Type de données : chaîne
- Type de valeur – À valeur unique

Cette clé de contexte est formatée selon le schéma `"aws:RequestTag/tag-key":"tag-value"`, où *`tag-key`* et *`tag-value`* représentent respectivement une clé de balise et une paire de valeurs. Les clés et les valeurs d'étiquette ne sont pas sensibles à la casse. Cela signifie que si vous spécifiez `"aws:RequestTag/TagKey1": "Value1"` dans l'élément de condition de votre politique, la condition correspond à une clé de balise de demande nommée `TagKey1` ou `tagkey1`, mais pas aux deux.

Cet exemple montre que même si la clé est à valeur unique, vous pouvez toujours utiliser plusieurs paires clé-valeur dans une requête si les clés sont différentes.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": "arn:aws:ec2:::instance/*",
    "Condition": {
      "StringEquals": {
        "aws:RequestTag/environment": [
          "preprod",
          "production"
        ],
        "aws:RequestTag/team": [
          "engineering"
        ]
      }
    }
  }
}
```

lois : TagKeys

Utilisez cette clé pour comparer les clés de balise d'une demande avec celles spécifiées dans la politique. Nous vous recommandons, lorsque vous utilisez des politiques pour contrôler l'accès à l'aide de balises, d'utiliser la clé de condition `aws:TagKeys` pour définir quelles clés de balises sont autorisées. Pour obtenir des exemples de stratégie et de plus amples informations, veuillez consulter the section called “Contrôle de l'accès en fonction des clés de balise”.

- Disponibilité : cette clé figure dans le contexte de la demande si l'opération prend en charge le passage de balises dans la demande.
- Type de données — Chaîne (liste)

- Type de valeur – À valeurs multiples

Cette clé de contexte est formatée selon le schéma `"aws:TagKeys":"tag-key"`, où *`tag-key`* est une liste de clés d'étiquette sans valeur (par exemple, `["Dept","Cost-Center"]`).

Étant donné que vous pouvez inclure plusieurs paires clé-valeur de balise dans une demande, le contenu de la demande peut être une demande à valeurs multiples. Dans ce cas, vous devez utiliser le opérateurs d'ensemble `ForAllValues` ou `ForAnyValue`. Pour plus d'informations, consultez Clés de contexte à valeurs multiples.

Certains services prennent en charge le balisage avec des opérations de ressource, comme la création, la modification ou de la suppression d'une ressource. Pour autoriser le balisage et les opérations sous la forme d'un seul appel, vous devez créer une politique qui inclut à la fois l'action de balisage et celle de modification de ressource. Vous pouvez ensuite utiliser la clé de condition `aws:TagKeys` pour imposer l'utilisation de clés de balise spécifiques dans la demande. Par exemple, pour limiter les balises lorsque quelqu'un crée un instantané Amazon EC2, vous devez inclure l'action de création `ec2:CreateSnapshot` et l'action de balisage `ec2:CreateTags` dans la politique. Pour consulter une politique applicable à ce scénario qui utiliseaws:TagKeys, consultez la section Création d'un instantané avec des balises dans le guide de l'utilisateur Amazon EC2.

lois : SecureTransport

Utilisez cette clé pour vérifier si une demande est envoyée avec SSL. Le contexte de la demande renvoie `true` ou `false`. Dans une politique, vous pouvez autoriser des actions spécifiques uniquement si la demande est envoyée avec SSL.

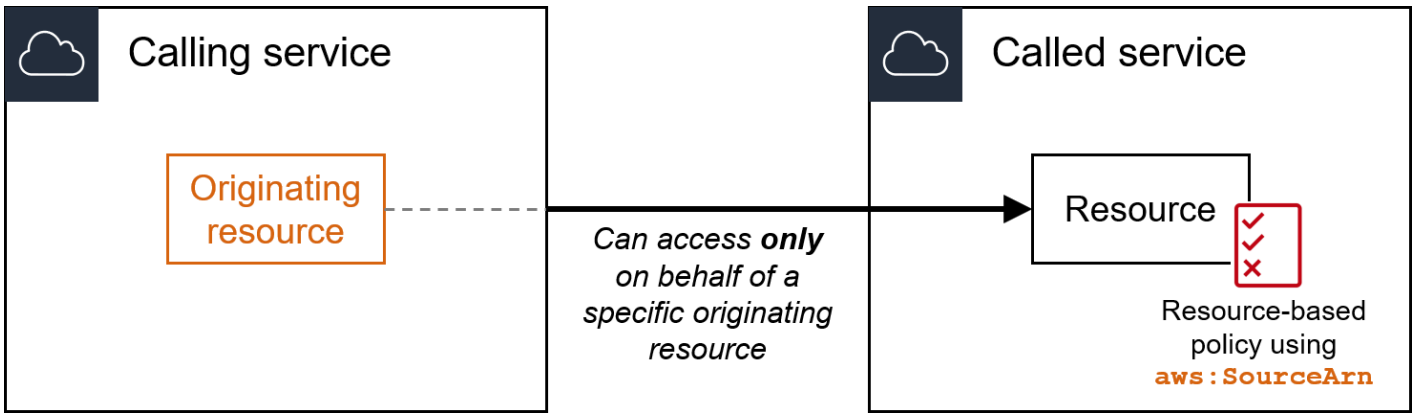
- Availability (Disponibilité) : cette clé figure toujours dans le contexte de la demande.
- Type de données : booléen
- Type de valeur – À valeur unique

lois : SourceArn

Utilisez cette clé pour comparer l'Amazon Resource Name (ARN) de la ressource qui fait une service-to-service demande avec l'ARN que vous spécifiez dans la politique, mais uniquement lorsque la demande est faite par un principal de AWS service. Lorsque l'ARN de la source inclut l'ID de compte, il n'est pas nécessaire d'utiliser `aws:SourceAccount` avec `aws:SourceArn`.

Cette clé ne fonctionne pas avec l'ARN du principal qui fait la demande. Utilisez à la place `lois : PrincipalArn`.

- Disponibilité — Cette clé est incluse dans le contexte de la demande uniquement lorsque l'appel à votre ressource est effectué directement par un principal de AWS service au nom d'une ressource pour laquelle la configuration a déclenché la service-to-service demande. Le service appelant transmet l'ARN de la ressource d'origine au service appelé.



Les intégrations de service suivantes ne prennent pas en charge cette clé de condition globale :

Service d'appel (principal du service)	Service appelé (politique basée sur une ressource)	Description
logdelivery.elb.amazonaws.com	Compartiment Amazon S3	Activer la journalisation des accès à Elastic Load Balancing dans le compartiment Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Compartiment Amazon S3	Activer la journalisation des accès à Elastic Load Balancing dans le compartiment Amazon S3

> **Note**
>
> Les intégrations de services avec AWS Security Token Service (AWS STS) et AWS Key Management Service (AWS KMS) ne sont pas toutes prises en charge. Pour plus d'informations, consultez la documentation du service appelant. L'utilisation de politiques

de clés intégrées `aws:SourceArn` au KMS pour les clés utilisées par le Services AWS biais de l'attribution de clés via KMS peut entraîner un comportement inattendu.

- Type de données : ARN, chaîne

  AWS recommande d'utiliser des opérateurs ARN plutôt que des opérateurs de chaîne lorsque vous comparez des ARN.

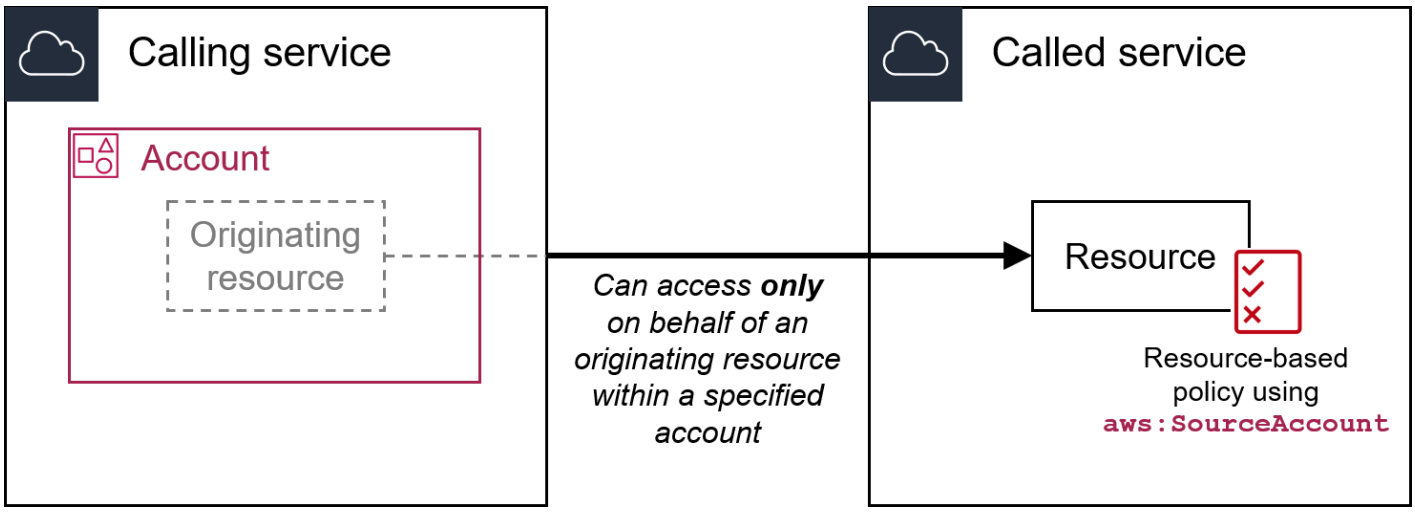
- Type de valeur – À valeur unique

Vous pouvez utiliser cette clé de condition pour éviter qu'un AWS service ne soit utilisé comme un adjoint confus lors de transactions entre services. Utilisez cette clé uniquement dans les politiques basées sur les ressources où il `Principal` s'agit d'un Service AWS principal. Définissez la valeur de cette clé de condition sur l'ARN de la ressource dans la demande. Par exemple, lorsqu'une mise à jour du compartiment Amazon S3 déclenche une publication de rubrique Amazon SNS, le service Amazon S3 appelle l'opération d'API `sns:Publish`. Dans la politique de rubrique qui autorise l'opération `sns:Publish`, définissez la valeur de la clé de condition sur l'ARN du compartiment Amazon S3. Pour savoir comment et quand cette clé de condition est recommandée, consultez la documentation des AWS services que vous utilisez.

lois : SourceAccount

Utilisez cette clé pour comparer l'ID de compte de la ressource qui fait une service-to-service demande avec l'ID de compte que vous spécifiez dans la politique, mais uniquement lorsque la demande est faite par un principal de AWS service.

- Disponibilité — Cette clé est incluse dans le contexte de la demande uniquement lorsque l'appel à votre ressource est effectué directement par un principal de AWS service au nom d'une ressource pour laquelle la configuration a déclenché la service-to-service demande. Le service appelant transmet l'ID de compte de la ressource au service appelé.



Les intégrations de service suivantes ne prennent pas en charge cette clé de condition globale :

Service d'appel (principal du service)	Service appelé (politique basée sur une ressource)	Description
logdelivery.elb.amazonaws.com	Compartiment Amazon S3	Activer la journalisation des accès à Elastic Load Balancing dans le compartiment Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Compartiment Amazon S3	Activer la journalisation des accès à Elastic Load Balancing dans le compartiment Amazon S3

> **Note**
>
> Les intégrations de services avec AWS Security Token Service (AWS STS) et AWS Key Management Service (AWS KMS) ne sont pas toutes prises en charge. Pour plus d'informations, consultez la documentation du service appelant. L'utilisation de politiques de clés intégrées `aws:SourceAccount` au KMS pour les clés utilisées par le Services AWS biais de l'attribution de clés via KMS peut entraîner un comportement inattendu.

- Type de données : chaîne

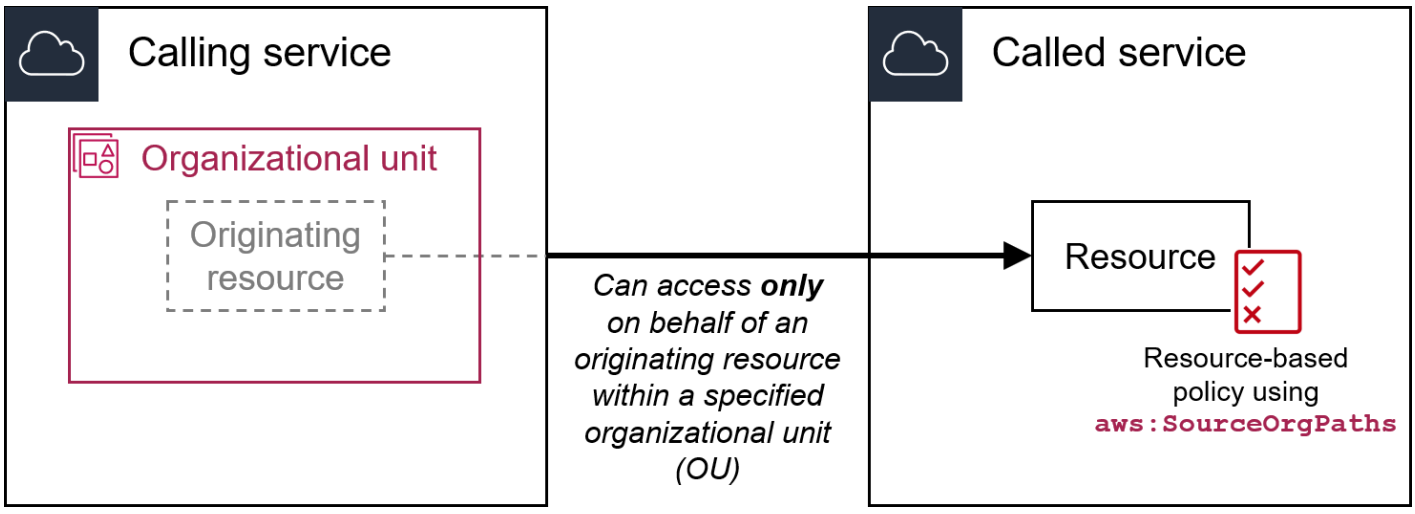
- Type de valeur – À valeur unique

Vous pouvez utiliser cette clé de condition pour éviter qu'un AWS service ne soit utilisé comme un adjoint confus lors de transactions entre services. Utilisez cette clé uniquement dans les politiques basées sur les ressources où il `Principal` s'agit d'un Service AWS principal. Définissez la valeur de cette clé de condition sur l'ID de compte de la ressource dans la demande. Par exemple, lorsqu'une mise à jour du compartiment Amazon S3 déclenche une publication de rubrique Amazon SNS, le service Amazon S3 appelle l'opération d'API `sns:Publish`. Dans la politique de rubrique qui autorise l'opération `sns:Publish`, définissez la valeur de la clé de condition sur l'ID de compte du compartiment Amazon S3. Pour savoir comment et quand ces clés de condition sont recommandées, consultez la documentation des AWS services que vous utilisez.

aws : SourceOrg Chemins

Utilisez cette clé pour comparer le AWS Organizations chemin de la ressource qui fait une service-to-service demande avec le chemin de l'organisation que vous spécifiez dans la politique, mais uniquement lorsque la demande est faite par un directeur de AWS service. Un chemin Organizations est une représentation textuelle de la structure d'une entité Organizations. Pour de plus amples informations sur l'utilisation et la compréhension des chemins, veuillez consulter Présentation du chemin d'entité AWS Organizations.

- Disponibilité : cette clé est contenue dans le contexte de la demande uniquement lorsque l'appel à votre ressource est effectué directement par un principal du service AWS au nom d'une ressource appartenant à un compte membre d'une organisation. Le service appelant doit transmet le chemin de l'organisation de la ressource d'origine au service appelé.



Les intégrations de service suivantes ne prennent pas en charge cette clé de condition globale :

Service d'appel (principal du service)	Service appelé (politique basée sur une ressource)	Description
logdelivery.elb.amazonaws.com	Compartiment Amazon S3	Activer la journalisation des accès à Elastic Load Balancing dans le compartiment Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Compartiment Amazon S3	Activer la journalisation des accès à Elastic Load Balancing dans le compartiment Amazon S3
Tous les principaux de service	Robot Amazon Lex	Services AWS Autoriser l'utilisation du bot Amazon Lex

> **Note**
>
> Les intégrations de services avec AWS Security Token Service (AWS STS) et AWS Key Management Service (AWS KMS) ne sont pas toutes prises en charge. Pour plus d'informations, consultez la documentation du service appelant. L'utilisation de politiques de clés intégrées `aws:SourceOrgPaths` au KMS pour les clés utilisées par le Services AWS biais de l'attribution de clés via KMS peut entraîner un comportement inattendu.

- Type de données — Chaîne (liste)
- Type de valeur – À valeur multiple

Vous pouvez utiliser cette clé de condition pour éviter qu'un AWS service ne soit utilisé comme un adjoint confus lors de transactions entre services. Utilisez cette clé uniquement dans les politiques basées sur les ressources où il `Principal` s'agit d'un Service AWS principal. Définissez la valeur de cette clé de condition sur le chemin de l'organisation de la ressource dans la demande. Par exemple, lorsqu'une mise à jour du compartiment Amazon S3 déclenche une publication de rubrique Amazon SNS, le service Amazon S3 appelle l'opération d'API `sns:Publish`. Dans la politique de rubrique qui autorise l'opération `sns:Publish`, définissez la valeur de la clé de condition sur le

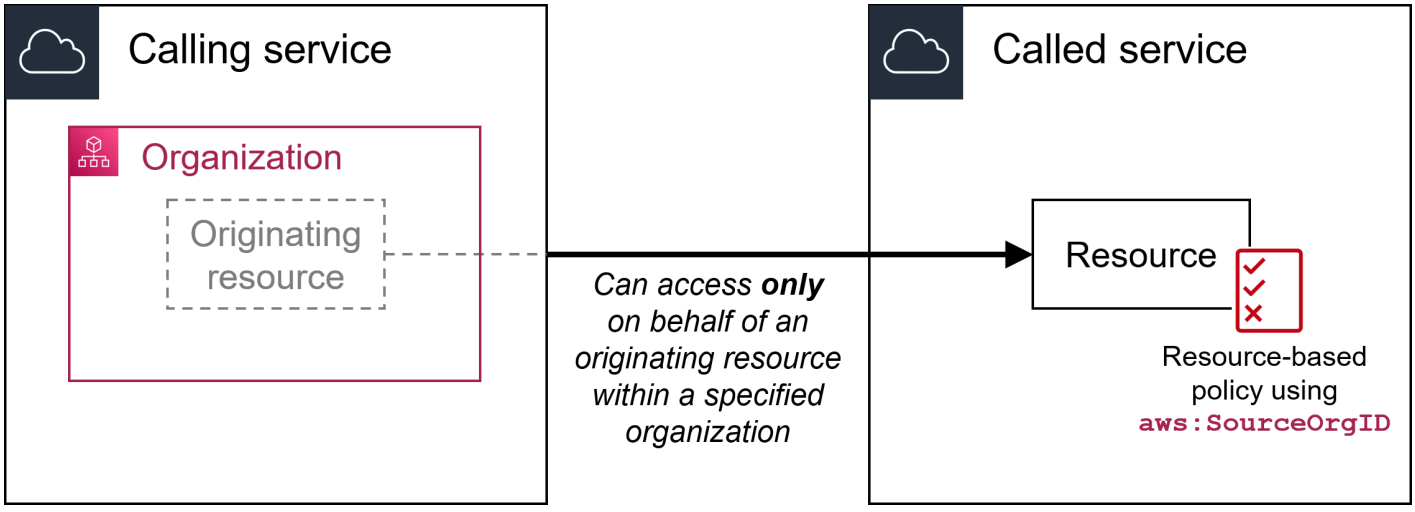
chemin de l'organisation du compartiment Amazon S3. Pour savoir comment et quand cette clé de condition est recommandée, consultez la documentation des AWS services que vous utilisez.

`aws:SourceOrgPaths` est une clé de condition à valeurs multiples. Les clés multivaluées peuvent avoir plusieurs valeurs dans le contexte de la demande. Vous devez utiliser les opérateurs d'ensemble `ForAnyValue` ou `ForAllValues` avec l'opérateur de condition de chaîne pour cette clé. Pour de plus amples informations sur les clés de condition à valeurs multiples, veuillez consulter Clés de contexte à valeurs multiples.

lois : SourceOrg ID

Utilisez cette clé pour comparer l'ID d'organisation de la ressource qui fait une service-to-service demande avec l'ID d'organisation que vous spécifiez dans la politique, mais uniquement lorsque la demande est faite par un directeur de AWS service. Lorsque vous ajoutez et supprimez des comptes à une organisation dans AWS Organizations, les politiques qui contiennent la clé `aws:SourceOrgID` incluent automatiquement les bons comptes et vous n'avez pas besoin de mettre à jour manuellement les politiques.

- Disponibilité : cette clé est contenue dans le contexte de la demande uniquement lorsque l'appel à votre ressource est effectué directement par un principal du service AWS au nom d'une ressource appartenant à un compte membre d'une organisation. Le service appelant transmet l'ID de l'organisation de la ressource d'origine au service appelé.



Les intégrations de service suivantes ne prennent pas en charge cette clé de condition globale :

Service d'appel (principal du service)	Service appelé (politique basée sur une ressource)	Description
logdelivery.elb.amazonaws.com	Compartiment Amazon S3	Activer la journalisation des accès à Elastic Load Balancing dans le compartiment Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Compartiment Amazon S3	Activer la journalisation des accès à Elastic Load Balancing dans le compartiment Amazon S3
Tous les principaux de service	Robot Amazon Lex	Services AWS Autoriser l'utilisation du bot Amazon Lex

> **Note**
>
> Les intégrations de services avec AWS Security Token Service (AWS STS) et AWS Key Management Service (AWS KMS) ne sont pas toutes prises en charge. Pour plus d'informations, consultez la documentation du service appelant. L'utilisation de politiques de clés intégrées `aws:SourceOrgID` au KMS pour les clés utilisées par le Services AWS biais de l'attribution de clés via KMS peut entraîner un comportement inattendu.

- Type de données : chaîne
- Type de valeur – À valeur unique

Vous pouvez utiliser cette clé de condition pour éviter qu'un AWS service ne soit utilisé comme un adjoint confus lors de transactions entre services. Utilisez cette clé uniquement dans les politiques basées sur les ressources où il `Principal` s'agit d'un Service AWS principal. Définissez la valeur de cette clé de condition sur l'ID de l'organisation de la ressource dans la demande. Par exemple, lorsqu'une mise à jour du compartiment Amazon S3 déclenche une publication de rubrique Amazon SNS, le service Amazon S3 appelle l'opération d'API `sns:Publish`. Dans la politique de rubrique qui autorise l'opération `sns:Publish`, définissez la valeur de la clé de condition sur l'ID de

l'organisation du compartiment Amazon S3. Pour savoir comment et quand cette clé de condition est recommandée, consultez la documentation des AWS services que vous utilisez.

lois : UserAgent

Utilisez cette clé pour comparer l'application cliente du demandeur avec l'application spécifiée dans la politique.

- Availability (Disponibilité) : cette clé figure toujours dans le contexte de la demande.
- Type de données : chaîne
- Type de valeur – À valeur unique

> **Warning**
>
> Utilisez cette clé avec précaution. Dans la mesure où le principal fournit la valeur `aws:UserAgent` dans un en-tête HTTP, les tiers non autorisés peuvent modifier ou personnaliser les navigateurs de manière à fournir n'importe quelle valeur `aws:UserAgent`. Par conséquent, `aws:UserAgent` il ne doit pas être utilisé pour empêcher des parties non autorisées de faire des AWS demandes directes. Vous pouvez l'utiliser pour autoriser uniquement les applications clientes spécifiques, et uniquement après avoir testé votre politique.

Autres clés de condition inter-services

AWS STS prend en charge les clés de condition de fédération basées sur SAML et les clés de condition interservices pour la fédération OIDC. Ces clés sont disponibles lorsqu'un utilisateur fédéré à l'aide de SAML effectue AWS des opérations dans d'autres services.

Clés de contexte IAM et de AWS STS condition

Vous pouvez utiliser `Condition` cet élément dans une politique JSON pour tester la valeur des clés incluses dans le contexte de toutes les AWS demandes. Ces clés fournissent des informations sur la demande elle-même ou les ressources référencées par cette dernière. Vous pouvez vérifier que les clés disposent de valeurs spécifiées avant d'autoriser l'action demandée par l'utilisateur. Vous bénéficiez ainsi d'un contrôle détaillé sur la correspondance ou non de vos instructions de politique JSON avec une demande entrante. Pour plus d'informations sur l'utilisation de l'élément `Condition` dans une politique JSON, consultez Éléments de politique JSON IAM : Condition.

Cette rubrique décrit les clés définies et fournies par le service IAM (avec un `iam:` préfixe) et le service AWS Security Token Service (AWS STS) (avec un `sts:` préfixe). Plusieurs autres AWS services fournissent également des clés spécifiques au service qui sont pertinentes pour les actions et les ressources définies par ce service. Pour plus d'informations, consultez Actions, ressources et clés de condition pour les AWS services. La documentation pour un service qui prend en charge des clés de condition possède généralement des informations supplémentaires. Par exemple, pour obtenir des informations sur les clés que pouvez utiliser dans les politiques relatives aux ressources Amazon S3, veuillez consulter clés de politique Amazon S3 dans le guide de l'utilisateur service de stockage simple Amazon.

Rubriques

- Clés disponibles pour IAM
- Clés disponibles pour la AWS fédération OIDC
- Clés disponibles pour la fédération SAML AWS STS
- Clés de contexte de fédération basées sur le protocole SAML AWS STS interservices
- Clés disponibles pour AWS STS

Clés disponibles pour IAM

Vous pouvez utilisez les clés de condition suivantes dans des politiques qui contrôlent l'accès aux ressources IAM :

iam : AssociatedResourceArn

Fonctionne avec des opérateurs ARN.

Spécifie l'ARN de la ressource à laquelle ce rôle sera associé dans le service de destination. La ressource appartient généralement au service auquel le principal transmet le rôle. Parfois, la ressource peut appartenir à un troisième service. Par exemple, vous pouvez transmettre à Amazon EC2 Auto Scaling un rôle qui sera utilisé sur une instance Amazon EC2. Dans ce cas, la condition correspondrait à l'ARN de l'instance Amazon EC2.

Cette clé de condition s'applique uniquement à l'PassRoleaction d'une politique. Elle ne peut pas être utilisée pour limiter une autre action.

Utilisez cette clé de condition dans une politique pour permettre à une entité de transmettre un rôle, mais uniquement si ce rôle est associé à la ressource spécifiée. Vous pouvez utiliser des

caractères génériques (*) pour autoriser des opérations effectuées sur un type spécifique de ressource sans restreindre la région ou l'ID de ressource. Par exemple, vous pouvez autoriser un utilisateur ou un rôle IAM à transmettre n'importe quel rôle au service Amazon EC2 à utiliser avec des instances de la région `us-east-1` ou `us-west-1`. L'utilisateur ou le rôle IAM ne serait pas autorisé à transmettre des rôles à d'autres services. De plus, Amazon EC2 n'est pas autorisé à utiliser le rôle avec des instances situées dans d'autres régions.

```
{
    "Effect": "Allow",
    "Action": "iam:PassRole",
    "Resource": "*",
    "Condition": {
        "StringEquals": {"iam:PassedToService": "ec2.amazonaws.com"},
        "ArnLike": {
            "iam:AssociatedResourceARN": [
                "arn:aws:ec2:us-east-1:111122223333:instance/*",
                "arn:aws:ec2:us-west-1:111122223333:instance/*"
            ]
        }
    }
}
```

> **Note**
>
> AWS services compatibles avec iam : prennent PassedToService également en charge cette clé de condition.

iam : AWSServiceName

Fonctionne avec des opérateurs de chaîne.

Spécifie le AWS service auquel ce rôle est attaché.

Dans cet exemple, vous autorisez une entité à créer un rôle lié à un service si le nom de service est access-analyzer.amazonaws.com.

```
{
   "Version": "2012-10-17",
   "Statement": [{
       "Effect": "Allow",
```

```
        "Action": "iam:CreateServiceLinkedRole",
        "Resource": "*",
        "Condition": {
          "StringLike": {
            "iam:AWSServiceName": "access-analyzer.amazonaws.com"
          }
        }
      }]
}
```

iam:FIDO-certification

Fonctionne avec des opérateurs de chaîne.

Vérifie le niveau de certification FIDO de l'appareil MFA au moment de l'enregistrement d'une clé de sécurité FIDO. La certification de l'appareil est extraite du FIDO Alliance Metadata Service (MDS). Si le statut ou le niveau de certification de votre clé de sécurité FIDO change, celui-ci ne sera pas mis à jour à moins que l'appareil ne soit désenregistré et enregistré de nouveau pour récupérer les informations de certification mises à jour.

Valeurs possibles de L1, L1plus, L2, L2plus, L3, L3plus

Dans cet exemple, vous enregistrez une clé de sécurité et récupérez la certification FIDO de niveau 1 plus pour votre appareil.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
```

```
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-certification": "L1plus"
                }
            }
        }
    ]

}
```

iam:FIDO-FIPS-140-2-certification

Fonctionne avec des opérateurs de chaîne.

Vérifie le niveau de certification de la validation FIPS-140-2 de l'appareil MFA au moment de l'enregistrement d'une clé de sécurité FIDO. La certification de l'appareil est extraite du FIDO Alliance Metadata Service (MDS). Si le statut ou le niveau de certification de votre clé de sécurité FIDO change, celui-ci ne sera pas mis à jour à moins que l'appareil ne soit désenregistré et enregistré de nouveau pour récupérer les informations de certification mises à jour.

Valeurs possibles de L1, L2, L3, L4

Dans cet exemple, vous enregistrez une clé de sécurité et récupérez la certification FIPS-140-2 de niveau 2 pour votre appareil.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
```

```
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        }
    ]

}
```

iam:FIDO-FIPS-140-3-certification

Fonctionne avec des opérateurs de chaîne.

Vérifie le niveau de certification de la validation FIPS-140-3 de l'appareil MFA au moment de l'enregistrement d'une clé de sécurité FIDO. La certification de l'appareil est extraite du FIDO Alliance Metadata Service (MDS). Si le statut ou le niveau de certification de votre clé de sécurité FIDO change, celui-ci ne sera pas mis à jour à moins que l'appareil ne soit désenregistré et enregistré de nouveau pour récupérer les informations de certification mises à jour.

Valeurs possibles de L1, L2, L3, L4

Dans cet exemple, vous enregistrez une clé de sécurité et récupérez la certification FIPS-140-3 de niveau 3 pour votre appareil.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L3"
```

```
                }
            }
        }
    ]

}
```

iam : RegisterSecurityKey

Fonctionne avec des opérateurs de chaîne.

Vérifie l'état actuel de l'activation de l'appareil MFA.

Valeurs possibles de `Create` ou `Activate`.

Dans cet exemple, vous enregistrez une clé de sécurité et récupérez la certification FIPS-140-3 de niveau 1 pour votre appareil.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L1"
                }
            }
        }
    ]
```

```
}
```

iam : OrganizationsPolicyId

Fonctionne avec des opérateurs de chaîne.

Vérifie que la politique avec l' AWS Organizations ID spécifié correspond à la politique utilisée dans la demande. Pour voir un exemple de politique IAM qui utilise cette clé de condition, veuillez consulter IAM : afficher les dernières informations consultées relatives au service pour une politique Organizations.

iam : PassedToService

Fonctionne avec des opérateurs de chaîne.

Spécifie le principal du service auquel un rôle peut être transmis. Cette clé de condition s'applique uniquement à l'PassRoleaction d'une politique. Elle ne peut pas être utilisée pour limiter une autre action.

Lorsque vous utilisez cette clé de condition dans une politique, spécifiez le service à l'aide d'un principal service. Un principal de service est le nom d'un service qui peut être spécifié dans l'élément `Principal` d'une politique. Il s'agit du format habituel :`SERVICE_NAME_URL.amazonaws.com`.

Vous pouvez utiliser `iam:PassedToService` pour limiter vos utilisateurs de sorte qu'ils puissent transmettre des rôles uniquement à des services spécifiques. Par exemple, un utilisateur peut créer un rôle de service qui l'autorise CloudWatch à écrire des données de journal dans un compartiment Amazon S3 en son nom. Ensuite, l'utilisateur doit attacher une politique d'autorisation et une politique d'approbation au nouveau rôle de service. Dans ce cas, la politique d'approbation doit spécifier `cloudwatch.amazonaws.com` dans l'élément `Principal`. Pour consulter une politique qui permet à l'utilisateur de transmettre le rôle CloudWatch, consultezIAM : transmettre un rôle IAM à un service spécifique AWS.

En utilisant cette clé de condition, vous pouvez vous assurer que les utilisateurs créent des rôles de service uniquement pour les services que vous spécifiez. Par exemple, si un utilisateur de la politique précédente tente de créer un rôle de service pour Amazon EC2, l'opération échouera. L'échec se produit, car l'utilisateur n'est pas autorisé à transmettre le rôle à Amazon EC2.

Parfois, vous transmettez un rôle à un service, qui le transmet ensuite à un service différent. `iam:PassedToService` inclut uniquement le service final qui endosse le rôle, et non le service intermédiaire qui transmet le rôle.

> **Note**
>
> Certains services ne prennent pas en charge cette clé de condition.

iam : PermissionsBoundary

Fonctionne avec des opérateurs ARN.

Vérifie que la politique spécifiée est attachée en tant que limite d'autorisations sur la ressource du principal IAM. Pour plus d'informations, veuillez consulter Limites d'autorisations pour les entités IAM

iam:PolicyARN

Fonctionne avec des opérateurs ARN.

Vérifie l'ARN (Amazon Resource Name) d'une politique gérée dans les demandes qui impliquent ce type de politique. Pour plus d'informations, consultez Contrôle de l'accès aux politiques.

iam :ResourceTag//nom-clé

Fonctionne avec des opérateurs de chaîne.

Vérifie que l'étiquette attachée à la ressource d'identité (utilisateur ou rôle) correspond aux nom et valeur de la clé spécifiée.

> **Note**
>
> IAM et AWS STS supporte à la fois la clé de condition `iam:ResourceTag` IAM et la clé de condition `aws:ResourceTag` globale.

Vous pouvez ajouter des attributs personnalisés à des ressources IAM sous la forme d'une paire clé-valeur. Pour plus d'informations sur les balises pour les ressources IAM, consultez the section called “Balisage des ressources IAM”. Vous pouvez utiliser `ResourceTag` pour contrôler l'accès aux ressources AWS , y compris les ressources IAM. Cependant, comme IAM ne prend pas en charge les balises pour les groupes, vous ne pouvez pas utiliser de balises pour contrôler l'accès aux groupes.

Cet exemple montre comment vous pouvez créer une politique basée sur l'identité qui autorise la suppression des utilisateurs avec la balise **`status=terminated`**. Pour utiliser cette politique,

remplacez le *texte de l'espace réservé en italique* dans l'exemple de politique par vos propres informations de ressource. Ensuite, suivez les instructions fournies dans create a policy (créer une politique) ou edit a policy (modifier une politique).

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:DeleteUser",
        "Resource": "*",
        "Condition": {"StringEquals": {"iam:ResourceTag/status": "terminated"}}
    }]
}
```

Clés disponibles pour la AWS fédération OIDC

Vous pouvez utiliser la fédération OIDC pour fournir des informations de sécurité temporaires aux utilisateurs authentifiés via un fournisseur d'identité compatible OpenID Connect (IdP) auprès d'un fournisseur d'identité IAM OpenID Connect (OIDC) dans votre compte. AWS Amazon Cognito GitHub, Login with Amazon et Google sont des exemples de tels fournisseurs. Des jetons d'identité et des jetons d'accès provenant de votre propre IdP peuvent être utilisés, ainsi que des jetons de compte de service accordés aux charges de travail Amazon Elastic Kubernetes Service.

Vous pouvez utiliser les clés contextuelles de condition AWS OIDC pour rédiger des politiques qui limitent l'accès des utilisateurs fédérés aux ressources associées à un fournisseur, une application ou un utilisateur spécifique. Ces clés sont généralement utilisées dans la politique d'approbation d'un rôle. Définissez les clés de condition en utilisant le nom du fournisseur OIDC (`token.actions.githubusercontent.com`) suivi d'une réclamation (`:aud`) :**`token.actions.githubusercontent.com:aud`**.

Certaines clés de condition de fédération OIDC peuvent être utilisées dans la session de rôle pour autoriser l'accès aux ressources. Si la valeur est Oui dans la colonne Disponible dans la session, vous pouvez utiliser ces clés de condition dans les politiques pour définir les accès des utilisateurs dans d'autres AWS services. Lorsqu'une réclamation n'est pas disponible en session, la clé contextuelle de condition OIDC ne peut être utilisée que dans le cadre d'une politique de confiance des rôles pour l'AssumeRoleWithWebIdentityauthentification initiale.

Sélectionnez votre IdP pour voir comment les réclamations de votre IdP sont associées au contexte des conditions IAM entrent en ligne de compte. AWS

Default

La valeur par défaut répertorie les revendications OIDC standard et la manière dont elles sont mappées aux clés de contexte de AWS STS condition. AWS Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés de AWS STS condition aux valeurs de la colonne de réclamation IdP JWT. Utilisez ce mappage si votre IdP n'est pas répertorié dans les options de l'onglet.

GitHub Les workflows Actions et Google en sont quelques exemples IdPs qui utilisent l'implémentation par défaut dans leur jeton d'identification OIDC JWT.

AWS STS clé de condition	Réclamation IdP JWT	Disponible en cours de session
amr	amr	Oui
aud	azp Si aucune valeur n'est définie`azp`, la clé de `aud` condition correspond à la `aud` réclamation.	Oui
e-mail	e-mail	Non
oaud	aud	Non
sub	sub	Oui

Pour plus d'informations sur l'utilisation des clés contextuelles de condition OIDC avec GitHub, consultezConfiguration d'un rôle pour le fournisseur d'identité GitHub OIDC. Pour de plus amples informations sur les champs Google `aud` et `azp`, veuillez consulter le guide Google Identity Platform OpenID Connect.

amr

Fonctionne avec des opérateurs de chaîne. La clé comporte plusieurs valeurs, ce qui signifie que vous l'analysez dans une politique à l'aide d'opérateurs de définition de condition.

Exemple : `token.actions.githubusercontent.com:amr`

La référence des méthodes d'authentification inclut les informations de connexion de l'utilisateur. Elle contient les valeurs suivantes :

- Si l'utilisateur n'est pas authentifié, la clé contient uniquement `unauthenticated`.
- Si l'utilisateur est authentifié, la clé contient la valeur `authenticated` et le nom du fournisseur de connexion utilisé dans l'appel (`accounts.google.com`).

aud

Fonctionne avec des opérateurs de chaîne.

Exemples :

- `accounts.google.com:aud`
- `token.actions.githubusercontent.com:aud`

Utilisez la clé de `aud` condition pour vérifier que le public correspond à celui que vous spécifiez dans la politique. Vous pouvez utiliser la clé aud avec la sous-clé du même fournisseur d'identité.

Cette clé de condition est définie à partir des champs de jeton suivants :

- `aud` pour les ID de client Google OAuth 2.0 de votre application, lorsque le champ `azp` n'est pas défini. Lorsque le champ `azp` est défini, le champ `aud` correspond à la clé de condition `accounts.google.com:oaud`.
- `azp` lorsque le champ `azp` est défini. Cela peut se produire pour les applications hybrides au sein desquelles une application web et une application Android ont un ID de client Google OAuth 2.0 différent mais partagent le même projet d'API Google.

Lorsque vous écrivez une politique à l'aide de la clé de condition `accounts.google.com:aud`, vous devez savoir si l'application est une application hybride qui définit le champ `azp`.

Champ non défini `azp`

L'exemple de politique suivant fonctionne pour les applications non hybrides qui ne définissent pas le champ `azp`. Dans ce cas, la valeur du champ de jeton d'ID Google `aud` correspond à la fois aux valeurs de clé de condition `accounts.google.com:aud` et `accounts.google.com:oaud`.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "aud-value",
                    "accounts.google.com:oaud": "aud-value",
                    "accounts.google.com:sub": "sub-value"
                }
            }
        }
    ]
}
```

Champ défini `azp`

L'exemple de politique suivant fonctionne pour les applications hybrides qui définissent le champ `azp`. Dans ce cas, la valeur du champ de jeton d'ID Google `aud` correspond uniquement à la valeur de clé de condition `accounts.google.com:oaud`. La valeur du champ `azp` correspond à la valeur de clé de condition `accounts.google.com:aud`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "azp-value",
                    "accounts.google.com:oaud": "aud-value",
                    "accounts.google.com:sub": "sub-value"
                }
            }
        }
    ]
}
```

email

Fonctionne avec des opérateurs de chaîne.

Exemple : `accounts.google.com:email`

Cette clé de condition valide l'adresse e-mail de l'utilisateur. La valeur de cette réclamation n'est peut-être pas propre à ce compte et peut changer au fil du temps. Par conséquent, vous ne devez pas utiliser cette valeur comme identifiant principal pour vérifier votre dossier d'utilisateur.

oaud

Fonctionne avec des opérateurs de chaîne.

Exemple : `accounts.google.com:oaud`

Cette clé indique à quel autre public (`aud`) ce jeton d'identification est destiné. Il doit s'agir de l'un des ID client OAuth 2.0 de votre application.

sub

Fonctionne avec des opérateurs de chaîne.

Exemples :

- `accounts.google.com:sub`
- token.actions.githubusercontent.com : sub

Utilisez ces clés pour vérifier que le sujet correspond à celui que vous spécifiez dans la politique. Vous pouvez utiliser la clé `sub` avec la clé `aud` pour le même fournisseur d'identité.

Dans la politique de confiance des rôles suivante, la clé de `sub` condition limite le rôle à la GitHib branche nomméedemo.

```
{
   "Version": "2012-10-17",
   "Statement": [
      "Condition": {
         "StringEquals": {
            "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
            "token.actions.githubusercontent.com:sub": "repo:octo-org/octo-
repo:ref:refs/heads/demo"
         }
      }
   ]
}
```

Amazon Cognito

Cet onglet explique comment Amazon Cognito mappe les allégations OIDC pour AWS STS conditionner les clés contextuelles. AWS Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés de AWS STS condition aux valeurs de la colonne de réclamation IdP JWT.

Pour les rôles utilisés par Amazon Cognito, les clés sont définies par l'URL `cognito-identity.amazonaws.com` suivie de la demande.

Pour plus d'informations sur le mappage des demandes relatives au pool d'identités, consultez la section Mappages des fournisseurs par défaut dans le manuel Amazon Cognito Developer Guide. Pour plus d'informations sur le mappage des réclamations du pool d'utilisateurs, consultez la section Utilisation du jeton d'identification dans le manuel Amazon Cognito Developer Guide.

AWS STS clé de condition	Réclamation IdP JWT	Disponible en cours de session
amr	amr	Oui
aud	aud	Oui
oaud	aud	Non
sub	sub	Oui

amr

Fonctionne avec des opérateurs de chaîne. La clé comporte plusieurs valeurs, ce qui signifie que vous l'analysez dans une politique à l'aide d'opérateurs de définition de condition.

Exemple — `cognito-identity.amazonaws.com:amr`

La référence des méthodes d'authentification inclut les informations de connexion de l'utilisateur. Elle contient les valeurs suivantes :

- Si l'utilisateur n'est pas authentifié, la clé contient uniquement `unauthenticated`.
- Si l'utilisateur est authentifié, la clé contient la valeur `authenticated` et le nom du fournisseur de connexion utilisé dans l'appel (`cognito-identity.amazonaws.com`).

Par exemple, la condition suivante de la politique de confiance pour un rôle Amazon Cognito vérifie si l'utilisateur n'est pas authentifié.

```
"Condition": {
  "StringEquals":
    { "cognito-identity.amazonaws.com:aud": "us-east-2:identity-pool-id" },
  "ForAnyValue:StringLike":
    { "cognito-identity.amazonaws.com:amr": "unauthenticated" }
}
```

aud

Fonctionne avec des opérateurs de chaîne.

Exemple — `cognito-identity.amazonaws.com:aud`

Client d'application du groupe d'utilisateurs qui a authentifié l'utilisateur. Amazon Cognito affiche la même valeur dans le champ standard `client_id` du jeton d'accès.

oaud

Fonctionne avec des opérateurs de chaîne.

Exemple — `cognito-identity.amazonaws.com:oaud`

Client d'application du groupe d'utilisateurs qui a authentifié l'utilisateur. Amazon Cognito affiche la même valeur dans le champ standard `client_id` du jeton d'accès.

sub

Fonctionne avec des opérateurs de chaîne.

Exemple — `cognito-identity.amazonaws.com:sub`

Identifiant unique (UUID), ou sujet, de l'utilisateur authentifié. Le nom d'utilisateur n'est peut-être pas unique dans votre groupe d'utilisateurs. La sous-réclamation est le meilleur moyen d'identifier un utilisateur donné. Vous pouvez utiliser la clé `sub` avec la clé `aud` pour le même fournisseur d'identité.

```
{
   "Version": "2012-10-17",
   "Statement": [
      "Condition": {
```

```
         "StringEquals": {
            "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-abcd-abcd-
abcd-123456790ab",
            "cognito-identity.amazonaws.com:sub": [
               "us-east-1:12345678-1234-1234-1234-123456790ab",
               "us-east-1:98765432-1234-1234-1243-123456790ab"
            ]
         }
      }
   ]
}
```

Login with Amazon

Cet onglet explique comment Login with Amazon mappe les affirmations de l'OIDC visant à AWS STS conditionner les clés contextuelles. AWS Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés de AWS STS condition aux valeurs de la colonne de réclamation IdP JWT.

AWS STS clé de condition	Réclamation IdP JWT	Disponible en cours de session
app_id	ID d'application	Oui
sub	ID de l'utilisateur	Oui
user_id	ID de l'utilisateur	Oui

identifiant de l'application

Fonctionne avec des opérateurs de chaîne.

Exemple — `www.amazon.com:app_id`

Cette clé indique le contexte d'audience qui correspond au `aud` champ utilisé par les autres fournisseurs d'identité.

sub

Fonctionne avec des opérateurs de chaîne.

Exemple — `www.amazon.com:sub`

Cette clé vérifie que l'ID utilisateur correspond à celui que vous spécifiez dans la politique. Vous pouvez utiliser la clé `sub` avec la clé `aud` pour le même fournisseur d'identité.

identifiant_utilisateur

Fonctionne avec des opérateurs de chaîne.

Exemple — `www.amazon.com:user_id`

Cette clé indique le contexte d'audience qui correspond au `aud` champ utilisé par les autres fournisseurs d'identité. Vous pouvez utiliser la `user_id` clé avec la `id` clé du même fournisseur d'identité.

Facebook

Cet onglet explique comment Facebook mappe les affirmations de l'OIDC pour AWS STS conditionner les clés contextuelles. AWS Vous pouvez utiliser ces clés pour contrôler l'accès à un rôle. Pour ce faire, comparez les clés de AWS STS condition aux valeurs de la colonne de réclamation IdP JWT.

AWS STS clé de condition	Réclamation IdP JWT	Disponible en cours de session
app_id	ID d'application	Oui
id	id	Oui

identifiant de l'application

Fonctionne avec des opérateurs de chaîne.

Exemple — `graph.facebook.com:app_id`

Cette clé vérifie que le contexte d'audience correspond au `aud` champ utilisé par les autres fournisseurs d'identité.

id

Fonctionne avec des opérateurs de chaîne.

Exemple — `graph.facebook.com:id`

Cette clé a vérifié que l'identifiant de l'application (ou du site) correspond à celui que vous spécifiez dans la politique.

Plus d'informations sur la fédération OIDC

- Guide de l'utilisateur d'Amazon Cognito
- Fédération OIDC

Clés disponibles pour la fédération SAML AWS STS

Si vous utilisez une fédération basée sur SAML à l'aide de AWS Security Token Service (AWS STS), vous pouvez inclure des clés de condition supplémentaires dans la politique.

Politiques d'approbation de rôle SAML

Dans la politique d'approbation d'un rôle, vous pouvez inclure les clés suivantes afin d'établir si le principal est autorisé à endosser le rôle. Hormis `saml:doc`, toutes les valeurs sont dérivées de l'assertion SAML. Tous les éléments de la liste sont disponibles dans l'éditeur visuel de la console IAM lorsque vous créez ou modifiez une politique avec des conditions. Les éléments marqués d'un `[]` peuvent avoir une valeur qui est une liste du type spécifié.

saml:aud

Fonctionne avec des opérateurs de chaîne.

L'URL d'un point de terminaison auquel les assertions SAML sont présentées. La valeur de cette clé provient du champ `SAML Recipient` de l'assertion, non du champ `Audience`.

saml:commonName[]

Fonctionne avec des opérateurs de chaîne.

Attribut `commonName`.

saml:cn[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduOrg`.

saml:doc

Fonctionne avec des opérateurs de chaîne.

Représente le principal utilisé pour endosser le rôle. Le format est *Account-ID*/*provider-friendly-name*, tel que. `123456789012/SAMLProviderName` La valeur ID compte fait référence au compte qui est propriétaire du fournisseur SAML.

saml:edupersonaffiliation[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonassurance[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonentitlement[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonnickname[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonorgdn

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonorgunitdn[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonprimaryaffiliation

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonprimaryorgunitdn

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonprincipalname

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersonscopedaffiliation[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:edupersontargetedid[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduPerson`.

saml:eduorghomepageuri[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduOrg`.

saml:eduorgidentityauthnpolicyuri[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduOrg`.

saml:eduorglegalname[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduOrg`.

saml:eduorgsuperioruri[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduOrg`.

saml:eduorgwhitepagesuri[]

Fonctionne avec des opérateurs de chaîne.

Attribut `eduOrg`.

saml:givenName[]

Fonctionne avec des opérateurs de chaîne.

Attribut `givenName`.

saml:iss

Fonctionne avec des opérateurs de chaîne.

Auteur, représenté par un URN.

saml:mail[]

Fonctionne avec des opérateurs de chaîne.

Attribut `mail`.

saml:name[]

Fonctionne avec des opérateurs de chaîne.

Attribut `name`.

saml:namequalifier

Fonctionne avec des opérateurs de chaîne.

Valeur de hachage basée sur le nom convivial du fournisseur SAML. La valeur est la concaténation des valeurs suivantes, dans l'ordre et séparées par le caractère '/' :

1. Valeur de la réponse `Issuer` (`saml:iss`)
2. ID du compte `AWS`.
3. Nom convivial (dernière partie de l'ARN) du fournisseur SAML dans IAM

La concaténation de l'ID de compte et du nom convivial du fournisseur SAML est disponible dans les politiques IAM en tant que clé `saml:doc`. Pour plus d’informations, veuillez consulterIdentification unique des utilisateurs dans la fédération SAML.

saml:organizationStatus[]

Fonctionne avec des opérateurs de chaîne.

Attribut `organizationStatus`.

saml:primaryGroupSID[]

Fonctionne avec des opérateurs de chaîne.

Attribut `primaryGroupSID`.

saml:sub

Fonctionne avec des opérateurs de chaîne.

Objet de la demande, qui inclut une valeur qui identifie de manière unique un utilisateur individuel au sein d'une organisation (par exemple, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

saml:sub_type

Fonctionne avec des opérateurs de chaîne.

Cette clé peut avoir la valeur `persistent`, `transient`, ou consister en l'URI `Format` complet des éléments `Subject` et `NameID` utilisés dans votre assertion SAML. La valeur `persistent` indique que la valeur de `saml:sub` reste la même pour l'utilisateur entre les sessions. Si la valeur est `transient`, l'utilisateur a une valeur `saml:sub` différente pour chaque session. Pour plus d'informations sur l'attribut `NameID` de l'élément `Format`, consultez Configurer les assertions SAML pour la réponse d'authentification.

saml:surname[]

Fonctionne avec des opérateurs de chaîne.

Attribut `surnameuid`.

saml:uid[]

Fonctionne avec des opérateurs de chaîne.

Attribut `uid`.

saml:x500 [] UniqueIdentifier

Fonctionne avec des opérateurs de chaîne.

Attribut `x500UniqueIdentifier`.

Pour obtenir des informations générales sur les attributs `eduPerson` et `eduOrg`, accédez au site Web EFEDS Wiki. Pour obtenir la liste des attributs `eduPerson`, consultez Spécifications des classes d'objets eduPerson (201602).

Les clés de condition dont le type est une liste peuvent inclure plusieurs valeurs. Pour créer des conditions dans la politique pour des valeurs de liste, vous pouvez utiliser des opérateurs de définition (`ForAllValues`, `ForAnyValue`). Par exemple, pour autoriser un utilisateur dont l'affiliation est « faculté » ou « personnel » (mais pas « étudiant »), vous pouvez utiliser la condition suivante :

```
"Condition": {
   "ForAllValues:StringLike": {
     "saml:edupersonaffiliation":[ "faculty", "staff"]
   }
}
```

Clés de contexte de fédération basées sur le protocole SAML AWS STS interservices

Certaines clés de condition de fédération basées sur SAML peuvent être utilisées dans des demandes ultérieures pour autoriser AWS des opérations dans d'autres services et `AssumeRole` appels. Il s'agit des clés de condition suivantes qui peuvent être utilisées dans les politiques de confiance des rôles lorsque les principaux fédérés assument un autre rôle, et dans les politiques de ressources d'autres AWS services pour autoriser l'accès aux ressources par des principaux fédérés. Pour plus d'informations sur l'utilisation de ces clés, consultez la section À propos de la fédération SAML 2.0.

Sélectionnez une clé de condition pour voir la description.

- saml:namequalifier
- saml:sub
- saml:sub_type

> **Note**
>
> Aucune autre clé de condition pour la fédération basée sur SAML n'est disponible à l'utilisation après la réponse d'authentification initiale du fournisseur d'identité (IdP) externe.

Clés disponibles pour AWS STS

Vous pouvez utiliser les clés de condition suivantes dans les politiques de confiance des rôles IAM pour les rôles assumés à l'aide d'opérations AWS Security Token Service (AWS STS).

saml:sub

Fonctionne avec des opérateurs de chaîne.

Objet de la demande, qui inclut une valeur qui identifie de manière unique un utilisateur individuel au sein d'une organisation (par exemple, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

sets : AWSServiceName

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour spécifier un service dans lequel un jeton porteur fonctionne. Lorsque vous utilisez cette clé de condition dans une politique, spécifiez le service à l'aide d'un principal service. Un principal de service est le nom d'un service qui peut être spécifié dans l'élément `Principal` d'une politique. Par exemple, `codeartifact.amazonaws.com` c'est le principal du AWS CodeArtifact service.

Certains AWS services nécessitent que vous soyez autorisé à obtenir un jeton de support de AWS STS service avant de pouvoir accéder à leurs ressources par programmation. Par exemple, AWS CodeArtifact exige des principaux qu'ils utilisent des jetons porteurs pour effectuer certaines opérations. La commande `aws codeartifact get-authorization-token` renvoie un jeton porteur. Vous pouvez ensuite utiliser le jeton porteur pour effectuer des AWS CodeArtifact opérations. Pour de plus amples informations sur les jetons porteurs, veuillez consulter Utilisation des jetons porteurs.

Disponibilité : cette clé est présente dans les demandes qui obtiennent un jeton porteur. Vous ne pouvez pas appeler directement pour AWS STS obtenir un jeton au porteur. Lorsque vous effectuez certaines opérations dans d'autres services, le service demande le jeton au porteur en votre nom.

Vous pouvez utiliser cette clé de condition pour permettre aux entités d'obtenir un jeton porteur à utiliser avec un service spécifique.

sets : DurationSeconds

Fonctionne avec des opérateurs numériques.

Utilisez cette clé pour spécifier la durée (en secondes) que le principal peut utiliser pour obtenir un jeton au AWS STS porteur.

Certains AWS services nécessitent que vous soyez autorisé à obtenir un jeton de support de AWS STS service avant de pouvoir accéder à leurs ressources par programmation. Par exemple, AWS CodeArtifact exige des principaux qu'ils utilisent des jetons porteurs pour effectuer certaines opérations. La commande `aws codeartifact get-authorization-token` renvoie un jeton porteur. Vous pouvez ensuite utiliser le jeton porteur pour effectuer des AWS CodeArtifact opérations. Pour de plus amples informations sur les jetons porteurs, veuillez consulter Utilisation des jetons porteurs.

Disponibilité : cette clé est présente dans les demandes qui obtiennent un jeton porteur. Vous ne pouvez pas appeler directement pour AWS STS obtenir un jeton au porteur. Lorsque vous effectuez certaines opérations dans d'autres services, le service demande le jeton au porteur en votre nom. La clé n'est pas présente pour les opérations AWS STS assume-role.

sets : ExternalId

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour exiger qu'un principal fournisse un identifiant spécifique lors de la prise en charge d'un rôle IAM.

Disponibilité — Cette clé est présente dans la demande lorsque le principal fournit un identifiant externe tout en assumant un rôle à l'aide de l' AWS API AWS CLI or.

Identifiant unique qui peut être requis lorsque vous endossez un rôle dans un autre compte. Si l'administrateur du compte auquel appartient le rôle vous a fourni un ID externe, indiquez cette valeur dans le paramètre `ExternalId`. Cette valeur peut être n'importe quelle chaîne, comme une phrase de passe ou un numéro de compte. La fonction principale de l'ID externe consiste à traiter et à prévenir le problème du député confus. Pour de plus amples informations sur l'ID externe et le problème du député confus, veuillez consulter Comment utiliser un identifiant externe lorsque vous accordez l'accès à vos AWS ressources à un tiers.

La valeur `ExternalId` peut avoir un minimum de 2 caractères et un maximum de 1 224 caractères. La valeur doit être alphanumérique sans espaces. Elle peut également inclure les symboles suivants : signe plus (+), signe égal (=), virgule (,), point (.), arobase (@), deux points (:), barre oblique (/) et tiret (-).

sts :RequestContext/touche contextuelle

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour comparer les paires clé-valeur du contexte de session intégrées dans l'assertion contextuelle signée par l'émetteur du jeton sécurisé transmise dans la demande avec les paires clé-valeurs spécifiées dans la politique de confiance des rôles.

Disponibilité — Cette clé est présente dans la demande lorsqu'une assertion de contexte est fournie dans le paramètre de `ProvidedContexts` demande tout en assumant un rôle à l'aide de l'opération AWS STS `AssumeRole` API.

Cette clé de contexte a le format suivant : `"sts:RequestContext/context-key":"context-value"` où `context-key` et `context-value` constituent une paire clé-valeur de contexte. Lorsque plusieurs clés de contexte sont intégrées dans l'assertion de contexte signée transmise dans la demande, il y a une clé de contexte pour chaque paire clé-valeur. Vous devez autoriser l'action `sts:SetContext` dans la politique de confiance des rôles afin de permettre à un principal de définir des clés de contexte dans le jeton de session obtenu. Pour en savoir plus sur les clés contextuelles IAM Identity Center prises en charge qui peuvent être utilisées avec cette clé, consultez la section clés de AWS STS condition pour IAM Identity Center dans le guide de l'AWS IAM Identity Center utilisateur.

Vous pouvez utiliser cette clé dans une politique de confiance des rôles pour appliquer un contrôle d'accès précis basé sur l'utilisateur ou ses attributs lorsqu'il endosse un rôle. Par exemple, vous pouvez configurer Amazon Redshift en tant qu'application IAM Identity Center pour accéder aux ressources Amazon S3 au nom de votre personnel ou des identités fédérées.

La politique de confiance des rôles suivante autorise le principal du service Amazon Redshift à endosser un rôle dans le compte `111122223333`. Il autorise également le principal du service Amazon Redshift à définir des clés de contexte dans la demande, à condition que la valeur de la clé de contexte `identitystore:UserId` soit définie sur `1111-22-3333-44-5555`. Une fois le rôle assumé, l'activité apparaît dans les AWS CloudTrail journaux de l'`AdditionalEventData`élément, contenant les paires clé-valeur du contexte de session définies par le fournisseur de contexte dans la demande d'attribution du rôle. Cela permet aux administrateurs de différencier plus facilement les sessions de rôle lorsqu'un rôle est utilisé par différents principaux. Les paires clé-valeur sont définies par le fournisseur de contexte spécifié, et non par AWS CloudTrail ou. AWS STS Cela permet au fournisseur de contexte de contrôler le contexte inclus dans les CloudTrail journaux et les informations de session.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Principal": {
                "Service": "redshift.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:SetContext"
            ],
            "Condition": {
                "ForAllValues:ArnEquals": {
                    "sts:RequestContextProviders": [
                        "arn:aws:iam::aws:contextProvider/IdentityCenter"
                    ]
                },
                "StringEquals": {
                    "aws:SourceAccount": "111122223333",
                    "sts:RequestContext/identitystore:UserId":
 "1111-22-3333-44-5555"
                }
            }
        }
    ]
}
```

sets : RequestContextProviders

Fonctionne avec des opérateurs ARN.

Utilisez cette clé pour comparer l'ARN du fournisseur de contexte dans la demande avec l'ARN du fournisseur de contexte spécifié dans la politique de confiance des rôles.

Disponibilité — Cette clé est présente dans la demande lorsqu'une assertion de contexte est fournie dans le paramètre de `ProvidedContexts` demande tout en assumant un rôle à l'aide de l'opération AWS STS `AssumeRole` API.

L'exemple de condition suivant vérifie que l'ARN du fournisseur de contexte transmis dans la demande correspond à l'ARN spécifié dans la condition de politique d'approbation des rôles.

```
      "Condition": {
        "ForAllValues:ArnEquals": {
          "sts:RequestContextProviders": [
            "arn:aws:iam::aws:contextProvider/IdentityCenter"
          ]
```

```
        }
      }
```

sets : RoleSessionName

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour comparer le nom de session spécifié par un principal lorsqu'il endosse un rôle avec la valeur spécifiée dans la politique.

Disponibilité — Cette clé est présente dans la demande lorsque le principal assume le rôle à l' AWS Management Console aide de toute commande CLI assume-role ou de toute opération d' AWS STS `AssumeRole`API.

Vous pouvez utiliser cette clé dans une politique d'approbation de rôle pour exiger que vos utilisateurs fournissent un nom de session spécifique lorsqu'ils endossent un rôle. Par exemple, vous pouvez exiger que les utilisateurs IAM spécifient leur propre nom d'utilisateur comme nom de session. Une fois que l'utilisateur IAM endosse le rôle, l'activité apparaît dans les journaux AWS CloudTrail sous le nom de session correspondant à son nom d'utilisateur. Cela permet aux administrateurs de différencier plus facilement les sessions de rôle lorsqu'un rôle est utilisé par différents principaux.

La politique d'approbation de rôle suivante exige que les utilisateurs IAM dans le compte `111122223333` fournissent leur nom d'utilisateur IAM comme nom de session lorsqu'ils endossent le rôle. Cette exigence est appliquée à l'aide de la variable de condition `aws:username` dans la clé de condition. Cette politique permet aux utilisateurs IAM d'endosser le rôle auquel la politique est attachée. Cette politique empêche quiconque utilise des informations d'identification temporaires d'endosser le rôle, car la variable `username` est uniquement présente pour les utilisateurs IAM.

> **Important**
>
> Vous pouvez utiliser n'importe quelle clé de condition à valeur unique comme variable. Vous ne pouvez pas utiliser de clé de condition à valeurs multiples en tant que variable.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "RoleTrustPolicyRequireUsernameForSessionName",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
            "Condition": {
                "StringLike": {"sts:RoleSessionName": "${aws:username}"}
            }
        }
    ]
}
```

Lorsqu'un administrateur consulte le AWS CloudTrail journal d'une action, il peut comparer le nom de session aux noms d'utilisateur de son compte. Dans l'exemple suivant, l'utilisateur appelé `matjac` a effectué l'opération en utilisant le rôle appelé `MateoRole`. L'administrateur peut alors contacter Mateo Jackson, dont le nom d'utilisateur est `matjac`.

```
    "assumedRoleUser": {
        "assumedRoleId": "AROACQRSTUVWRAOEXAMPLE:matjac",
        "arn": "arn:aws:sts::111122223333:assumed-role/MateoRole/matjac"
    }
```

Si vous autorisez l'accès entre comptes à l'aide de rôles, les utilisateurs d'un compte peuvent endosser un rôle dans un autre compte. L'ARN de l'utilisateur du rôle assumé répertorié dans la liste CloudTrail inclut le compte sur lequel le rôle existe. Il n'inclut pas le compte de l'utilisateur qui a endossé le rôle. Les utilisateurs ne sont uniques qu'au sein d'un compte. Par conséquent, nous vous recommandons d'utiliser cette méthode pour vérifier les CloudTrail journaux uniquement pour les rôles assumés par les utilisateurs dans les comptes que vous administrez. Vos utilisateurs peuvent utiliser le même nom d'utilisateur dans plusieurs comptes.

sets : SourceIdentity

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour comparer l'identité source spécifiée par un principal lorsqu'il endosse un rôle avec la valeur spécifiée dans la politique.

Disponibilité — Cette clé est présente dans la demande lorsque le principal fournit une identité source tout en assumant un rôle à l'aide d'une commande AWS STS CLI assume-role ou d'une opération d' AWS STS `AssumeRole`API.

Vous pouvez utiliser cette clé dans une politique d'approbation de rôle pour exiger que vos utilisateurs définissent une identité source spécifique lorsqu'ils endossent un rôle. Par exemple,

vous pouvez exiger que votre main-d'œuvre ou vos identités fédérées spécifient une valeur pour l'identité source. Vous pouvez configurer votre fournisseur d'identité (IdP) de sorte qu'il utilise l'un des attributs associés à vos utilisateurs, un nom d'utilisateur ou un e-mail par exemple, comme identité source. L'IdP transmet ensuite l'identité source en tant qu'attribut dans les assertions ou les revendications auxquelles il envoie. AWS La valeur de l'attribut d'identité source identifie l'utilisateur ou l'application qui endosse le rôle.

Une fois que l'utilisateur endosse le rôle, l'activité apparaît dans les journaux AWS CloudTrail avec la valeur d'identité source qui a été définie. Cela permet aux administrateurs de déterminer plus facilement qui ou quoi a effectué des actions avec un rôle dans AWS. Vous devez octroyer des autorisations pour l'action `sts:SetSourceIdentity` afin d'autoriser une identité à définir une identité source.

Contrairement à `sts:RoleSessionName`, une fois l'identité source définie, la valeur ne peut plus être modifiée. Elle est présente dans le contexte de la demande pour toutes les actions effectuées avec le rôle par l'identité source. La valeur persiste dans les sessions de rôle suivantes lorsque vous utilisez les informations d'identification de session pour endosser un autre rôle. Le fait d'endosser un rôle à partir d'un autre est appelé chaînage des rôles.

Vous pouvez utiliser la clé de condition `aws:SourceIdentity`globale pour contrôler davantage l'accès aux AWS ressources en fonction de la valeur de l'identité de la source dans les demandes suivantes.

La politique de confiance de rôle suivante autorise l'utilisateur IAM `AdminUser`à endosser un rôle dans le compte `111122223333`. Elle octroie également l'autorisation à l'interface `AdminUser` de définir une identité source, dans la mesure où l'identité source définie correspond à `DiegoRamirez`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAdminUserAssumeRole",
            "Effect": "Allow",
            "Principal": {"AWS": " arn:aws:iam::111122223333:user/AdminUser"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
```

```
                "StringEquals": {"sts:SourceIdentity": "DiegoRamirez"}
            }
        }
    ]
}
```

Pour en savoir plus sur les informations relatives à l'identité source, veuillez consulter Surveiller et contrôler les actions prises avec les rôles endossés.

sets : TransitiveTagKeys

Fonctionne avec des opérateurs de chaîne.

Utilisez cette clé pour comparer les clés des balises de session transitoire de la demande avec celles spécifiées dans la politique.

Availability (Disponibilité) : cette clé est présente dans la demande lorsque vous effectuez une demande à l'aide d'informations d'identification de sécurité temporaires. Il s'agit notamment des informations d'identification créées à l'aide d'une opération assume-rôle, ou de l'opération `GetFederationToken`.

Lorsque vous effectuez une demande à l'aide d'informations d'identification de sécurité temporaires, le contexte de la demande inclut la clé de contexte `aws:PrincipalTag`. Cette clé inclut une liste de balises de session, de balises de session transitoire et de balises de rôle. Les balises de session transitoires sont des balises qui perdurent pendant toutes les sessions ultérieures lorsque vous utilisez les informations d'identification de session pour endosser un autre rôle. Le fait d'endosser un rôle à partir d'un autre est appelé chaînage des rôles.

Vous pouvez utiliser cette clé de condition dans une politique pour exiger la définition de balises de session spécifiques comme transitoires lorsque vous endossez un rôle ou fédérez un utilisateur.

Actions, ressources et clés de condition pour les AWS services

Chaque AWS service peut définir des actions, des ressources et des clés de contexte de condition à utiliser dans les politiques IAM. Pour obtenir la liste des AWS services et de leurs actions, ressources et clés contextuelles de condition, consultez la section Actions, ressources et clés de condition dans la référence d'autorisation de service.

Ressources pour en savoir plus sur IAM

IAM est un produit riche, et vous trouverez de nombreuses ressources pour vous aider à en savoir plus sur la manière dont IAM peut vous aider à sécuriser vos ressources Compte AWS et vos ressources.

Rubriques

- Identités
- Informations d'identification (mots de passe, clés d'accès et dispositifs MFA)
- Autorisations et politiques
- Fédération et délégation
- IAM et autres produits AWS
- Pratiques de sécurité générales
- Ressources générales

Identités

Consultez les ressources suivantes pour en savoir plus sur la création, la gestion et l'utilisation d'identités.

- Gestion des identités dans IAM Identity Center : informations procédurales relatives à la création d'utilisateurs et de groupes dans IAM Identity Center.
- Identités IAM (utilisateurs, groupes d'utilisateurs et rôles): discussion approfondie sur les utilisateurs, les groupes et les rôles.

Informations d'identification (mots de passe, clés d'accès et dispositifs MFA)

Consultez les guides suivants pour gérer les mots de passe, les clés d'accès et les dispositifs MFA pour vos utilisateurs Compte AWS et pour ceux d'IAM.

- Gestion des mots de passe utilisateur dans AWS : décrit les options disponibles pour la gestion des mots de passe des utilisateurs IAM de votre compte.

- Gestion des clés d'accès pour les utilisateurs IAM : décrit le fonctionnement des clés d'accès et la manière dont vous pouvez les utiliser pour effectuer des appels programmatiques à AWS. Il existe d'autres alternatives plus sécurisées aux clés d'accès que nous vous recommandons d'envisager d'abord. Pour plus d'informations, consultez la section Considérations et alternatives relatives aux clés d'accès à long terme dans le Guide de Références générales AWS .
- Utilisation de l'authentification multifactorielle (MFA) dans AWS : décrit comment configurer votre compte et les utilisateurs IAM de manière à exiger à la fois un mot de passe et un code à usage unique qui est généré sur un appareil avant que la connexion soit autorisée. (Ceci est parfois appelé authentification à deux facteurs.)

Pour obtenir des informations générales sur les types d'informations d'identification que vous utilisez pour accéder à Amazon Web Services, consultez Informations d'identification de sécuritéAWS dans le Guide de Références générales AWS .

Autorisations et politiques

Découvrez le fonctionnement interne des politiques IAM, ainsi que les meilleures façons d'octroyer des autorisations :

- Politiques et autorisations dans IAM : présente le langage de politique utilisé pour définir les autorisations. Décrit comment les autorisations peuvent être attachées à des utilisateurs ou des groupes ou, dans le cas de certains produits AWS , aux ressources proprement dites.
- Références des éléments de politique JSON IAM : fournit des descriptions et des exemples de chaque élément de langage de politique.
- Validation de politiques IAM – recherche de ressources pour la validation de la politique JSON.
- Exemples de politiques basées sur l'identité IAM— Affiche des exemples de politiques relatives à des tâches courantes dans divers AWS produits.
- AWS Policy Generator (Générateur de politiques AWS) : permet de créer des politiques personnalisées en choisissant des produits et des actions dans une liste.
- Simulateur de politique IAM — Testez si une politique autoriserait ou refuserait une demande spécifique à AWS.

Fédération et délégation

Vous pouvez accorder l'accès aux ressources de votre site Compte AWS aux utilisateurs authentifiés (connectés) ailleurs. Il peut s'agir d'utilisateurs IAM appartenant à une autre Compte AWS entité (ce que l'on appelle la délégation), d'utilisateurs authentifiés par le biais du processus de connexion de votre organisation ou d'utilisateurs d'un fournisseur d'identité Internet tel que Login with Amazon, Facebook, Google ou tout autre fournisseur d'identité compatible avec OpenID Connect (OIDC). Dans ces cas, les utilisateurs obtiennent des informations d'identification de sécurité temporaires pour accéder aux AWS ressources.

- Didacticiel IAM : déléguer l'accès entre des comptes AWS à l'aide des rôles IAM : vous guide pour octroyer un accès intercompte à un utilisateur IAM dans un autre Compte AWS.
- Scénarios courants d'informations d'identification temporaires— Décrit les manières dont les utilisateurs peuvent être fédérés AWS après avoir été authentifiés en dehors de. AWS

IAM et autres produits AWS

La plupart des AWS produits sont intégrés à IAM afin que vous puissiez utiliser les fonctionnalités IAM pour protéger l'accès aux ressources de ces produits. Les ressources suivantes traitent de l'IAM et de la sécurité pour certains des AWS produits les plus populaires. Pour une liste complète des produits fonctionnant avec IAM, avec des liens vers des informations complémentaires sur chacun de ces produits, consultez AWS services qui fonctionnent avec IAM.

Utilisation d'IAM avec Amazon EC2

- Contrôle de l'accès aux ressources Amazon EC2 : décrit comment utiliser les fonctions IAM pour permettre aux utilisateurs de gérer des instances Amazon EC2, des volumes, et plus encore.
- Utilisation de profils d'instance— Décrit comment utiliser les rôles IAM pour fournir en toute sécurité des informations d'identification aux applications qui s'exécutent sur des instances Amazon EC2 et qui ont besoin d'accéder à AWS d'autres produits.

Utilisation d'IAM avec Amazon S3

- Gestion des autorisations d'accès à vos ressources Amazon S3 : présente le modèle de sécurité Amazon S3 pour les compartiments et les objets, notamment les politiques IAM.

- Writing IAM Policies: Grant Access to User-Specific Folders in an Amazon S3 Bucket : billet de blog qui décrit comment permettre aux utilisateurs de protéger leurs propres dossiers dans Amason S3. (Pour plus de billets se rapportant à Amazon S3 et IAM, sélectionnez la balise S3 sous le titre du billet de blog.)

Utilisation d'IAM avec Amazon RDS

- Utilisation de AWS Identity and Access Management (IAM) pour gérer l'accès aux ressources Amazon RDS : décrit comment utiliser IAM pour contrôler l'accès aux instances de base de données, aux instantanés de base de données, etc.
- A Primer on RDS Resource-Level Permissions : billet de blog qui décrit comment utiliser IAM pour contrôler l'accès à des instances Amazon RDS spécifiques.

Utilisation d'IAM avec Amazon DynamoDB

- Utilisation d'IAM pour contrôler l'accès aux ressources DynamoDB : décrit comment utiliser IAM pour permettre aux utilisateurs de gérer les tables et les index DynamoDB.
- La vidéo suivante (08:55) explique comment fournir un contrôle d'accès pour des éléments de base de données ou des attributs DynamoDB individuels (ou les deux).

  Getting Started with Fine-Grained Access Control for DynamoDB

Pratiques de sécurité générales

Trouvez des conseils et des conseils d'experts sur les meilleurs moyens de sécuriser vos ressources Compte AWS et vos ressources :

- Meilleures pratiques en matière de sécurité, d'identité et de conformité : trouvez des ressources sur la façon de gérer la sécurité de l'ensemble Comptes AWS des produits, notamment des suggestions concernant l'architecture de sécurité, l'utilisation de l'IAM, le chiffrement et la sécurité des données, etc.
- Identity and Access Management — The AWS Well-Architected Framework vous aide à comprendre les concepts clés, les principes de conception et les meilleures pratiques architecturales pour concevoir et exécuter des charges de travail dans le cloud.

- Bonnes pratiques de sécurité dans IAM : fournit des recommandations sur les différentes façons d'utiliser IAM pour sécuriser votre Compte AWS et vos ressources.
- AWS CloudTrail Guide de l'utilisateur : AWS CloudTrail permet de suivre l'historique des appels d'API effectués AWS et de stocker ces informations dans des fichiers journaux. Vous pouvez ainsi identifier les utilisateurs et les comptes ayant accédé aux ressources de votre compte lors des appels, déterminer les actions demandées, etc.

Ressources générales

Explorez les ressources suivantes pour en savoir plus sur IAM et AWS.

- Informations sur le produit IAM : informations générales sur le produit AWS Identity and Access Management .
- AWS re:Post pour AWS Identity and Access Management — Visite AWS re:Post pour discuter de questions techniques liées à l'IAM avec la AWS communauté.

- Cours et ateliers — Liens vers des cours spécialisés et basés sur des rôles, ainsi que des ateliers à votre rythme pour vous aider à perfectionner vos AWS compétences et à acquérir une expérience pratique.
- AWS Centre pour développeurs : découvrez les didacticiels, téléchargez des outils et découvrez les événements AWS destinés aux développeurs.
- AWS Outils de développement : liens vers des outils de développement, des SDK, des boîtes à outils IDE et des outils de ligne de commande pour le développement et la gestion AWS d'applications.
- Centre de ressources pour la mise en route : découvrez comment configurer votre application Compte AWS, rejoindre la AWS communauté et lancer votre première application.
- Tutoriels pratiques — Suivez les step-by-step didacticiels pour lancer votre première application sur AWS.
- AWS Livres blancs : liens vers une liste complète de livres AWS blancs techniques, traitant de sujets tels que l'architecture, la sécurité et l'économie, rédigés par des architectes de AWS solutions ou d'autres experts techniques.
- AWS Support Centre — Le centre de création et de gestion de vos AWS Support dossiers. Comprend également des liens vers d'autres ressources utiles, telles que des forums, des FAQ techniques, l'état de santé du service et AWS Trusted Advisor.

- AWS Support— La principale page Web contenant des informations sur AWS Support un one-on-one canal d'assistance à réponse rapide pour vous aider à créer et à exécuter des applications dans le cloud.
- Contactez-nous : point de contact central pour toute question relative à la facturation AWS , à votre compte, aux événements, à des abus ou à d'autres problèmes.
- AWS Conditions du site — Informations détaillées sur nos droits d'auteur et notre marque commerciale ; votre compte, votre licence et l'accès au site ; et d'autres sujets.

Appel de l'API IAM à l'aide de requêtes HTTP

Table des matières

- Points de terminaison
- HTTPS requis
- Signature des demandes d'API IAM

Vous pouvez accéder à l'IAM et aux AWS STS services par programmation à l'aide de l'API Query. Les demandes d'API de requête sont des demandes HTTPS qui doivent comporter un paramètre `Action` qui indique l'action à effectuer. IAM et AWS STS supporte les requêtes GET et POST pour toutes les actions. Autrement dit, l'API ne requiert pas l'utilisation de GET pour certaines actions et de POST pour d'autres. Toutefois, les requêtes GET sont soumises aux restrictions de taille applicables à une URL ; si cette limite varie en fonction du navigateur, elle est généralement de 2 048 octets. Par conséquent, dans le cas de demandes d'API de requête requérant des tailles plus importantes, il convient d'utiliser une requête POST.

Vous obtenez une réponse sous la forme d'un document XML. Pour plus d'informations sur la réponse, veuillez consulter les pages des actions spécifiques dans la Référence d'API IAM ou la Référence d'API AWS Security Token Service.

> **Tip**
>
> Au lieu d'appeler directement les opérations IAM ou AWS STS API, vous pouvez utiliser l'un des AWS SDK. Les AWS SDK se composent de bibliothèques et d'exemples de code pour différents langages de programmation et plateformes (Java, Ruby, .NET, iOS, Android, etc.). Les SDK constituent un moyen pratique de créer un accès programmatique à IAM et. AWS Par exemple, ils automatisent les tâches telles que la signature cryptographique des demandes (voir ci-dessous), la gestion des erreurs et les nouvelles tentatives de demande. Pour plus d'informations sur AWS les SDK, notamment sur la façon de les télécharger et de les installer, consultez la page Outils pour Amazon Web Services.

Pour obtenir des détails sur les actions d'API et les erreurs, veuillez consulter la Référence des API IAM ou la Référence des API AWS Security Token Service.

Points de terminaison

IAM et AWS STS chacun d'entre eux ont un point de terminaison global unique :

- (IAM) https://iam.amazonaws.com
- (AWS STS) https://sts.amazonaws.com

> **Note**
>
> AWS STS prend également en charge l'envoi de demandes aux points de terminaison régionaux en plus du point de terminaison global. Avant de pouvoir l'utiliser AWS STS dans une région, vous devez d'abord activer STS dans cette région pour votre Compte AWS. Pour plus d'informations sur l'activation de régions supplémentaires pour AWS STS, consultezGérer AWS STS dans un Région AWS.

Pour plus d'informations sur les AWS points de terminaison et les régions de tous les services, consultez la section Points de terminaison et quotas des services dans le. Références générales AWS

HTTPS requis

Dans la mesure où l'API de requête retourne des informations sensibles telles que les informations d'identification de sécurité, vous devez utiliser HTTPS avec toutes les demandes d'API.

Signature des demandes d'API IAM

Les demandes doivent être signées à l'aide d'un identifiant de la clé d'accès et d'une clé d'accès secrète. Nous vous recommandons fortement de ne pas utiliser les informations d'identification de votre Utilisateur racine d'un compte AWS pour l'exécution des tâches quotidiennes avec IAM. Vous pouvez utiliser les informations d'identification d'un utilisateur IAM ou vous pouvez les utiliser AWS STS pour générer des informations d'identification de sécurité temporaires.

Pour signer vos demandes d'API, nous vous recommandons d'utiliser AWS Signature Version 4. Pour plus d'informations sur l'utilisation de Signature Version 4, veuillez consulter Processus de signature Signature Version 4 dans le document Référence générale AWS .

Si vous devez utiliser Signature Version 2, des informations sur son utilisation sont disponibles dans le document Référence générale AWS.

Pour plus d'informations, veuillez consulter les ressources suivantes :

- AWS Identifiants de sécurité. Fournit des informations générales sur les types d'informations d'identification utilisés pour l'accès AWS.
- Bonnes pratiques de sécurité dans IAM. Présente une liste de suggestions pour utiliser le service IAM afin de sécuriser vos AWS ressources.
- Informations d'identification de sécurité temporaires dans IAM. Décrit comment créer et utiliser les informations d'identification de sécurité temporaires.

Historique du document pour IAM

Le tableau suivant décrit les principales mises à jour de la documentation pour IAM.

Modification	Description	Date
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	IAM Access Analyzer a ajouté la prise en charge de l'autorisation de récupérer des informations sur les politiques relatives aux utilisateurs et aux rôles IAM aux autorisations de niveau de service de Policy. AccessAnalyzer ServiceRole	30 mai 2024
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	IAM Access Analyzer a ajouté la prise en charge de l'autorisation de récupérer l'état actuel du blocage de l'accès public pour les instantanés Amazon EC2 aux autorisations de niveau de service de Policy. AccessAnalyzer ServiceRole	23 janvier 2024
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	IAM Access Analyzer a ajouté des flux et des tables DynamoDB aux autorisations de niveau de service de Policy. AccessAnalyzer ServiceRole	11 janvier 2024
AccessAnalyzerServiceRolePolicy : autorisations ajoutées	IAM Access Analyzer a ajouté des compartiments d'annuaire Amazon S3 aux autorisations de niveau de service de Policy. AccessAnalyzer ServiceRole	1er décembre 2023

IAMAccessAnalyzerReadOnlyAccess : autorisations ajoutées	IAM Access Analyzer a ajouté des autorisations à IAM AccessAnalyzer ReadOnly Access pour vous permettre de vérifier si les mises à jour de vos politiques accordent un accès supplémentaire. L'analyseur d'accès IAM a besoin de cette autorisation pour effectuer des vérifications de politique sur vos politiques.	26 novembre 2023
L’analyseur d’accès IAM a ajouté des analyseurs d’accès non utilisés	L’analyseur d’accès IAM simplifie l’inspection des accès non utilisés pour vous guider vers le moindre privilège . L’analyseur d’accès IAM analyse en permanence vos comptes pour identifier les accès non utilisés et crée un tableau de bord centralisé contenant les résultats.	26 novembre 2023
L’analyseur d’accès IAM a ajouté des vérifications de politiques personnalisées	L’analyseur d’accès IAM fournit désormais des vérifications de politiques personnalisées pour valider que les politiques IAM sont conformes à vos normes de sécurité avant les déploiements.	26 novembre 2023

AccessAnalyzerServiceRolePolicy : autorisations ajoutées	IAM Access Analyzer a ajouté des actions IAM aux autorisations de niveau de service de AccessAnalyzerServiceRolePolicy pour prendre en charge les actions suivantes : • Lister les entités pour une politique • Générer les dernières informations consultées sur le service • Lister les informations clés d'accès	26 novembre 2023
Prise en charge des informations de la dernière action consultée de l'élaboration de politiques pour plus de 60 services et actions supplémentaires	IAM prend désormais en charge les informations de la dernière action consultée et génère des politiques contenant des informations au niveau de l'action pour plus de 60 services supplémentaires, ainsi qu'une liste des actions pour lesquelles les informations de la dernière action consultée sont disponibles.	1er novembre 2023
Prise en charge des informations de la dernière action consultée pour plus de 140 services	IAM fournit désormais les informations de la dernière action consultée pour plus de 140 services, ainsi qu'une liste des actions pour lesquelles les informations de la dernière action consultée sont disponibles.	14 septembre 2023

Prise en charge des dispositifs d'authentification multifactorielle (MFA) pour les utilisateurs root et IAM	Vous pouvez désormais ajouter jusqu'à huit dispositifs MFA par utilisateur, y compris des clés de sécurité FIDO, un mot de passe à usage unique basé sur le temps (TOTP) associé à des applications d'authentification virtuelle ou des jetons TOTP matériels.	16 novembre 2022
Prise en charge de nouveaux types de ressources IAM Access Analyzer	L'IAM Access Analyzer a ajouté la prise en charge les types de ressources suivants : • Instantanés de volumes Amazon EBS • Référentiels Amazon ECR • Système de fichiers Amazon EFS • Instantanés de base de données Amazon RDS • Instantanés du cluster de base de données Amazon RDS • Rubriques Amazon SNS	25 octobre 2022
Obsolète U2F et mise à jour de /FIDO WebAuthn	Suppression des mentions d'U2F en tant qu'option MFA et ajout d'informations sur WebAuthn les clés de sécurité FIDO2 et FIDO.	31 mai 2022

Mises à jour de la résilience dans IAM	Ajout d'informations sur le maintien de l'accès aux informations d'identification IAM lorsqu'un événement perturbe la communication entre les Régions AWS.	16 mai 2022
Nouvelles clés de condition globales pour les ressources	Vous pouvez désormais contrôler l'accès aux ressources en fonction du compte, de l'unité organisationnelle (UO) ou de l'organisation AWS Organizations qui contient vos ressources. Vous pouvez utiliser les clés de condition globales `aws:ResourceAccount`, `aws:ResourceOrgID` et `aws:ResourceOrgPaths` dans une politique IAM.	27 avril 2022
Exemples de code pour IAM à l'aide AWS de kits de développement logiciel	Ajout d'exemples de code qui montrent comment utiliser IAM avec un kit de développement AWS logiciel (SDK). Les exemples sont divisés en extraits de code qui vous montrent comment appeler des fonctions de service individuelles et en exemples qui vous montrent comment accomplir une tâche spécifique en appelant plusieurs fonctions au sein d'un même service.	7 avril 2022

Mise à jour de l'organigramme logique d'évaluation des politiques	Mises à jour de l'organigramme logique d'évaluation des politiques et du texte associé dans la section déterminer si une demande est autorisée ou rejetée dans un compte.	17 novembre 2021
Mises à jour des bonnes pratiques de sécurité	Ajout d'informations sur la création d'utilisateurs administratifs au lieu d'utiliser les informations d'identification de l'utilisateur root, suppression des bonnes pratiques consistant à utiliser des groupes d'utilisateurs pour attribuer des autorisations aux utilisateurs IAM, et clarification des cas dans lesquels il convient d'utiliser des politiques gérées au lieu de politiques en ligne.	5 octobre 2021
Mises à jour de la rubrique logique d'évaluation des politiques pour les stratégies basées sur les ressources	Ajout d'informations sur l'impact des politiques basées sur les ressources et des différents types de principaux dans le même compte.	5 octobre 2021
Mises à jour des clés de condition à valeur unique et à valeurs multiples	Les différences entre les clés de condition à valeur unique et à valeurs multiples sont maintenant expliquées plus en détail. Le type de valeur a été ajouté à chaque clé de contexte de condition globale AWS.	30 septembre 2021

IAM Access Analyzer prend en charge des points d'accès multi-régions d'Amazon S3	IAM Access Analyzer identifie les compartiments Amazon S3 qui autorisent l'accès public et entre comptes, notamment ceux qui utilisent les points d'accès multi-régions Amazon S3.	2 septembre 2021
AWS mises à jour de politiques gérées - Mise à jour d'une politique existante	IAM Access Analyzer a mis à jour une politique AWS gérée existante.	2 septembre 2021
Davantage de services pris en charge pour la génération de politiques au niveau action	IAM Access Analyzer peut générer des politiques IAM avec des informations sur les activités d'accès au niveau de l'action pour des services supplémentaires. AWS	24 août 2021
Générer des politiques IAM pour des journaux d'activité entre comptes	Vous pouvez désormais utiliser IAM Access Analyzer pour générer des politiques précises en fonction de votre activité d'accès à l'aide d'un suivi AWS CloudTrail dans un autre compte, par exemple un suivi centralisé. AWS Organizations	18 août 2021

Vérifications de politiques IAM Access Analyzer supplémentaires	IAM Access Analyzer a étendu la validation de politique en ajoutant de nouvelles vérifications de politique qui valident les conditions incluses dans les politiques IAM. Ces vérifications analysent le bloc de condition dans votre instruction de politique et signalent les avertissements de sécurité, les erreurs et les suggestions, ainsi que des recommandations exploitables. IAM Access Analyzer a ajouté les vérifications de politique suivantes : • Erreur : format de principal de service non valide • Erreur : clé de balise manquante dans la condition • Avertissement de sécurité — Refuser NotAction avec étiquette non prise en charge, clé de condition pour le service • Avertissement de sécurité : refuser avec une clé de condition de balise non prise en charge pour le service • Avertissement de sécurité : clés de condition appariées manquantes	29 juin 2021

	• Suggestion — Autoriser NotAction avec étiquette non prise en charge, clé de condition pour le service • Suggestion : autoriser avec la clé de condition de balise non prise en charge pour le service	
Prise en charge de dernière action consultée, pour davantage de services	Vous pouvez désormais afficher des informations relatives à la dernière action consultée dans la console IAM, à propos de la dernière fois où un principal IAM a utilisé une action pour les services suivants : actions de gestion Amazon EC2, IAM, Lambda et Amazon S3. Vous pouvez également utiliser l' AWS API AWS CLI or pour récupérer un rapport de données. Elles vous permetten t d'identifier toute autorisat ion inutile. Vous pouvez ainsi peaufiner vos politiques IAM afin qu'elles respectent au plus près le principe du moindre privilège.	19 avril 2021

Surveiller et contrôler les mesures prises avec les rôles endossés	Les administrateurs peuvent configurer les rôles IAM pour exiger des identités qu'elles transmettent une identité source, qui est journalisée dans AWS CloudTrail. L'examen des informations d'identité source aide les administrateurs à déterminer qui ou ce qui a effectué des actions avec les sessions de rôle endossées.	13 avril 2021
Générer des politiques IAM basées sur l'activité d'accès	Vous pouvez désormais utiliser IAM Access Analyzer pour générer des politiques affinées en fonction de l'activité d'accès trouvée dans votre AWS CloudTrail.	7 avril 2021
Vérifications de politiques IAM Access Analyzer	IAM Access Analyzer fournit désormais plus de 100 vérifications de politiques avec des recommandations exploitables lors de la création de politiques.	16 mars 2021
Options de validation de politique étendues	Validation des politiques étendue disponible dans la console IAM, l' AWS API, et à AWS CLI l'aide des contrôles de politique dans IAM Access Analyzer pour vous aider à créer des politiques JSON sécurisées et fonctionnelles.	15 mars 2021

Balisage des ressources IAM	Vous pouvez maintenant baliser des ressources IAM supplémentaires à l'aide d'une paire clé-valeur de balise.	11 février 2021
Politique de mot de passe par défaut des utilisateurs IAM	Si vous ne définissez pas de politique de mot de passe personnalisée pour votre Compte AWS, les mots de passe des utilisateurs IAM doivent désormais respecter la politique de AWS mot de passe par défaut.	18 novembre 2020
Les pages d'actions, de ressources et de clés de condition pour les AWS services ont été déplacées	Chaque AWS service peut définir des actions, des ressources et des clés de contexte de condition à utiliser dans les politiques IAM. Vous pouvez désormais trouver la liste des AWS services et de leurs actions, ressources et clés contextuelles de condition dans la référence d'autoris ation des services.	16 novembre 2020

Durée de session de rôle plus longue des utilisateurs IAM	Les utilisateurs IAM peuvent désormais bénéficier d'une durée de session de rôle plus longue lorsqu'ils changent de rôle dans le AWS Management Console, ce qui réduit les interruptions dues à l'expiration des sessions. Les utilisateurs se voient accorder la durée maximale de session définie pour le rôle, ou la durée restante de la session de l'utilisateur IAM, selon la valeur la moins élevée.	24 juillet 2020
Utilisez Service Quotas pour demander des augmentations rapides pour les entités IAM	Vous pouvez demander des augmentations de quota pour des quotas IAM ajustables à l'aide de la console Service Quotas. Maintenant, certaines augmentations sont automatiquement approuvées dans Service Quotas et disponibles dans votre compte en quelques minutes. Les demandes plus importantes sont soumises à AWS Support.	25 juin 2020

Les dernières informations consultées dans IAM incluent maintenant les actions de gestion Amazon S3	Outre les données de dernier accès au service, vous pouvez désormais afficher des informations dans la console IAM concernant la dernière fois qu'un principal IAM a utilisé une action Amazon S3. Vous pouvez également utiliser l' AWS API AWS CLI or pour récupérer le rapport de données. Le rapport contient des informations sur les services autorisés et les actions auxquels les principaux ont tenté d'accéder pour la dernière fois et quand. Elles vous permettent d'identifier toute autorisation inutile. Vous pouvez ainsi peaufiner vos politiques IAM afin qu'elles respectent au plus près le principe du moindre privilège.	3 juin 2020
Ajout du chapitre sur la sécurité	Le chapitre sur la sécurité vous aide à comprendre comment configurer l'IAM et AWS STS à atteindre vos objectifs de sécurité et de conformité. Vous apprendrez également à utiliser d'autres services AWS pour surveiller et sécuriser vos ressources IAM.	29 avril 2020

sts : RoleSession Nom	Vous pouvez désormais écrire une politique qui accorde des autorisations en fonction du nom de session spécifié par un principal lors de l'attribution d'un rôle.	21 avril 2020
AWS mise à jour de la page de connexion	Lorsque vous vous connectez sur la page de AWS connexion principale, vous ne pouvez pas choisir de vous connecter en tant qu'utilisateur Utilisateur racine d'un compte AWS ou en tant qu'utilisateur IAM. Lorsque vous le faites, l'étiquette sur la page indique si vous devez fournir votre adresse e-mail d'utilisateur root ou les informations de votre utilisateur IAM. Cette documentation inclut des captures d'écran mises à jour pour vous aider à comprendre les pages de connexion AWS .	4 mars 2020

AWS:via AWSService et aws : CalledVia clés de condition	Vous pouvez désormais écrire une politique pour limiter le fait que les services puissent effectuer des demandes au nom d'un principal IAM (utilisateur ou rôle). Lorsqu'un principal effectue une demande à un service AWS , ce service peut utiliser les informations d'identification du principal pour effectuer des demandes ultérieures à d'autres services. Utilisez la clé de condition `aws:ViaAWSService` pour déterminer si un service effectue une demande à l'aide des informations d'identif ication d'un principal. Utilisez la clé de condition `aws:CalledVia` pour déterminer si des services spécifiques effectuen t une demande à l'aide des informations d'identification d'un principal.	20 février 2020
Ajout de la prise en charge des limites d'autorisations par le simulateur de politiques	Vous pouvez désormais tester l'effet des limites d'autoris ations sur les entités IAM à l'aide du simulateur de politique IAM.	23 janvier 2020

Évaluation de la politique entre comptes	Vous pouvez maintenant apprendre comment AWS évaluer les politiques d'accès entre comptes. Cette opération se produit lorsqu'une ressource d'un compte d'approbation inclut une politique basée sur la ressource permettant à un principal d'un autre compte d'accéder à la ressource. La demande doit être autorisée dans les deux comptes.	2 janvier 2020
Balises de session	Vous pouvez désormais inclure des balises lorsque vous endossez un rôle ou fédérez un utilisateur dans AWS STS. Lorsque vous effectuez les opérations `GetFederationToken` ou `AssumeRole` , vous pouvez transmettre les balises de session en tant qu'attributs. Lorsque vous effectuez les `AssumeRoleWithWebIdentity` opérations `AssumeRoleWithSAML` ou, vous pouvez transmettre des attributs de votre identité d'entreprise à AWS.	22 novembre 2019

Contrôlez l'accès pour les groupes Comptes AWS d'utilisateurs AWS Organizations	Vous pouvez désormais référencer les unités organisationnelles (UO) AWS Organizations à partir des politiques IAM. Si vous utilisez Organizations pour organiser vos comptes en unités d'organisation, vous pouvez exiger que les principaux appartiennent à une unité d'organisation spécifique avant d'accorder l'accès à vos ressources. Les principaux comprennent Utilisateur racine d'un compte AWS, les utilisateurs IAM et les rôles IAM. Pour ce faire, indiquez le chemin d'unité d'organisation dans la clé de condition `aws:PrincipalOrgPaths` de vos politiques.	20 novembre 2019
Dernier rôle utilisé	Vous pouvez désormais afficher la date, l'heure et la région auxquelles un rôle a été utilisé pour la dernière fois. Ces informations vous aident également à identifier les rôles inutilisés dans votre compte. Vous pouvez utiliser l' AWS Management Console AWS API AWS CLI et pour consulter les informations relatives à la date à laquelle un rôle a été utilisé pour la dernière fois.	19 novembre 2019

Mise à jour de la page des clés de contexte de condition globale	Vous pouvez désormais savoir quand chacune des clés de condition globale est incluse dans le contexte d'une requête. Vous pouvez également accéder à chaque clé plus facilement à l'aide de la table des matières de la page. Les informations sur la page vous aident à rédiger des politiques plus précises. Par exemple, si vos employés utilisent la fédération avec des rôles IAM, vous devez utiliser la clé `aws:userId` et non la clé `aws:userName` . La clé `aws:userName` s'applique uniquement aux utilisateurs IAM et non aux rôles.	6 octobre 2019

ABAC en AWS	Découvrez comment le contrôle d'accès basé sur les attributs (ABAC) fonctionn e à AWS l'aide de balises et comment il se compare au modèle d'autorisation tradition nel AWS . Utilisez le didactici el ABAC pour apprendre à créer et tester une politique qui permet aux rôles IAM avec des balises principales d'accéder aux ressources avec des balises correspondantes. Cette stratégie permet aux individus de consulter ou de modifier uniquement les AWS ressources nécessaires à leur travail.	3 octobre 2019
AWS STS GetAccessKeyInfo opération	Vous pouvez vérifier les clés AWS d'accès contenues dans votre code pour détermine r si elles proviennent d'un compte que vous possédez. Vous pouvez transmettre un identifiant de clé d'accès à l'aide de la `aws sts get-access-key-info` AWS CLI commande ou de l'opérati on `GetAccessKeyInfo` AWS API.	24 juillet 2019

Affichage des dernières informations consultées sur le service Organizations dans IAM	Vous pouvez désormais consulter les informations du dernier accès au service pour une AWS Organizations entité ou une politique dans la AWS Organizationssection de la console IAM. Vous pouvez également utiliser l' AWS API AWS CLI or pour récupérer le rapport de données. Ces données incluent les informations concernant les services autorisés auxquels les principaux d'un compte Organizations ont tenté d'accéder pour la dernière fois et quand. Elles vous permettent d'identifier toute autorisation inutile. Vous pouvez ainsi peaufiner vos politiques Organizations afin qu'elles respectent au plus près le principe du moindre privilège.	20 juin 2019
Utilisation d'une politique gérée en tant que politique de session	Vous pouvez désormais transmettre jusqu'à 10 ARN de politique gérée lorsque vous endossez un rôle. Ceci vous permet de limiter les autorisat ions des informations d'identif ication temporaires du rôle.	7 mai 2019

AWS STS Compatibilité régionale des jetons de session pour le point de terminaison global	Vous pouvez désormais choisir d'utiliser la version 1 ou la version 2 des jetons du point de terminaison global. Les jetons de version 1 ne sont valides que dans les AWS régions disponibles par défaut. Ces jetons ne fonctionnent pas dans les régions activées manuellement, par exemple Asie-Pacifique (Hong Kong). Les jetons de la version 2 sont valides dans toutes les régions. Toutefois, les jetons de la version 2 sont plus longs et peuvent avoir un impact sur les systèmes où vous stockez temporairement les jetons.	26 avril 2019
Autoriser l'activation et la désactivation des régions AWS	Vous pouvez désormais créer une politique qui autorise un administrateur à activer et désactiver la région Asie-Pacifique (Hong Kong) (ap us-east-1).	24 avril 2019

Page My Security Credentials (Mes informations d'identification de sécurité) de l'utilisateur IAM	Les utilisateurs IAM peuvent désormais gérer leurs propres informations d'identification sur la page My Security Credentials (Mes informations d'identification de sécurité). Cette AWS Management Console page affiche les informations du compte, telles que l'identifiant du compte et l'identifiant utilisateur canonique. Les utilisateurs peuvent également consulter et modifier leurs propres mots de passe, clés d'accès, certificats X.509, clés SSH et informations d'identification Git.	24 janvier 2019
API Access Advisor	Vous pouvez désormais utiliser l' AWS API AWS CLI and pour consulter les dernières informations du service auxquelles vous avez accédé.	7 décembre 2018

Balisage des utilisateurs et des rôles IAM	Vous pouvez désormais utiliser les balises IAM pour ajouter des attributs personnalisés à une identité (utilisateur ou rôle IAM) à l'aide d'une paire de balises clé-valeur. Vous pouvez également utiliser des balises pour contrôler l'accès aux ressources d'une identité ou pour contrôler les balises qui peuvent être attachées à une identité.	14 novembre 2018
Clés de sécurité U2F	Vous pouvez désormais utiliser les clés de sécurité U2F comme option d'authentification multi-facteurs (MFA, Multi-Factor Authentication) lors de la connexion à la AWS Management Console.	25 septembre 2018
Prise en charge des points de terminaison Amazon VPC	Vous pouvez désormais établir une connexion privée entre votre VPC et la région de AWS STS l'ouest des États-Unis (Oregon).	31 juillet 2018
Limites d'autorisations	La nouvelle fonctionnalité facilite les employés approuvés à accorder la possibilité de gérer les autorisations IAM sans également accorder un accès administratif IAM complet.	12 juillet 2018

lois : PrincipalOrg ID	La nouvelle clé de condition permet de contrôler plus facilement l'accès aux AWS ressources en spécifian t l' AWS organisation des principaux IAM.	17 mai 2018
lois : RequestedRegion	La nouvelle clé de condition permet d'utiliser plus facilemen t les politiques IAM pour contrôler l'accès aux AWS régions.	25 avril 2018
Augmentation de la durée de la session pour les rôles IAM	Un rôle IAM peut désormais avoir une durée de session de 12 heures.	28 mars 2018
Mise à jour du flux de travail pour la création de rôles	Un nouveau flux de travail améliore le processus de création des relations d'approbation et d'attachement des autorisations aux rôles.	8 septembre 2017
Compte AWS processus de connexion	L'expérience de AWS connexion mise à jour permet à l'utilisateur root et aux utilisateurs IAM d'utiliser le lien Se connecter à la console sur la page d'accueil AWS Management Console de l'utilisateur.	25 août 2017
Exemple de politiques IAM	La mise à jour de la documentation comprend plus de 30 exemples de politiques.	2 août 2017

Bonnes pratiques IAM	Les informations ajoutées dans la section Utilisateurs de la console IAM facilitent le respect des bonnes pratiques IAM.	5 juillet 2017
Ressources Auto Scaling	Les autorisations au niveau des ressources peuvent contrôler l'accès aux ressources Auto Scaling et les autorisations pour ces dernières.	16 mai 2017
Bases de données Amazon RDS for MySQL et Amazon Aurora	Les administrateurs de base de données peuvent associer les utilisateurs de base de données aux utilisateurs et aux rôles IAM et ainsi gérer l'accès des utilisateurs à toutes les AWS ressources à partir d'un seul emplacement.	24 avril 2017
Rôles liés à un service	Les rôles liés aux services constituent un moyen plus simple et plus sûr de déléguer des autorisations aux AWS services.	19 avril 2017
Récapitulatifs de la politique	De nouveaux récapitulatifs de la politique permettent de comprendre plus simplement les autorisations dans les politiques IAM.	23 mars 2017

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.