aws

Guide du développeur

Amazon Simple Queue Service

Amazon Simple Queue Service: Guide du développeur

Table of Contents

Qu'est-ce qu'Amazon Simple Queue Service

Amazon Simple Queue Service (Amazon SQS) offre une file d'attente hébergée sécurisée, durable et disponible qui vous permet d'intégrer et de découpler les systèmes et les composants de logiciels distribués. Amazon SQS propose des structures communes, telles que des files d'attente de lettres mortes et des balises de répartition des coûts. Il fournit une API de services Web générique à laquelle vous pouvez accéder à l'aide de n'importe quel langage de programmation pris en charge par le AWS SDK.

Rubriques

Avantages offerts par l'utilisation d'Amazon SQS

- Sécurité : vous pouvez contrôler qui peut envoyer et recevoir des messages à partir d'une file d'attente Amazon SQS. Vous pouvez choisir de transmettre des données sensibles en protégeant le contenu des messages dans les files d'attente grâce au chiffrement côté serveur (SSE) géré par Amazon SQS par défaut ou grâce aux clés SSE personnalisées gérées dans AWS Key Management Service (AWS KMS).
- Durabilité : pour assurer la sécurité de vos messages, Amazon SQS les stocke sur plusieurs serveurs. Les files d'attente standard prennent en charge la livraison des at-least-once messages, tandis que les files d'attente FIFO prennent en charge le traitement des messages en une seule fois et le mode haut débit.
- Disponibilité : Amazon SQS utilise une infrastructure redondante pour fournir un accès extrêmement simultané aux messages, et une haute disponibilité pour la production et la consommation des messages.
- Capacité de mise à l'échelle : Amazon SQS peut traiter chaque demande mise en mémoire tampon de façon indépendante, en se mettant à l'échelle de manière transparente pour gérer les augmentations et les pics de charge sans aucune instruction de mise en service.
- Fiabilité : Amazon SQS verrouille vos messages pendant le traitement pour que plusieurs producteurs puissent envoyer des messages et que plusieurs consommateurs puissent recevoir des messages en même temps.

- Personnalisation : vos files d'attente n'ont pas à être parfaitement identiques. Par exemple, vous pouvez définir un retard par défaut sur une file d'attente. Vous pouvez stocker le contenu des messages d'une taille supérieure à 256 Ko avec Amazon Simple Storage Service (Amazon S3) ou Amazon DynamoDB, avec Amazon SQS pour maintenir un pointeur vers l'objet Amazon S3. Vous pouvez également fractionner un message volumineux en messages de plus petite taille.

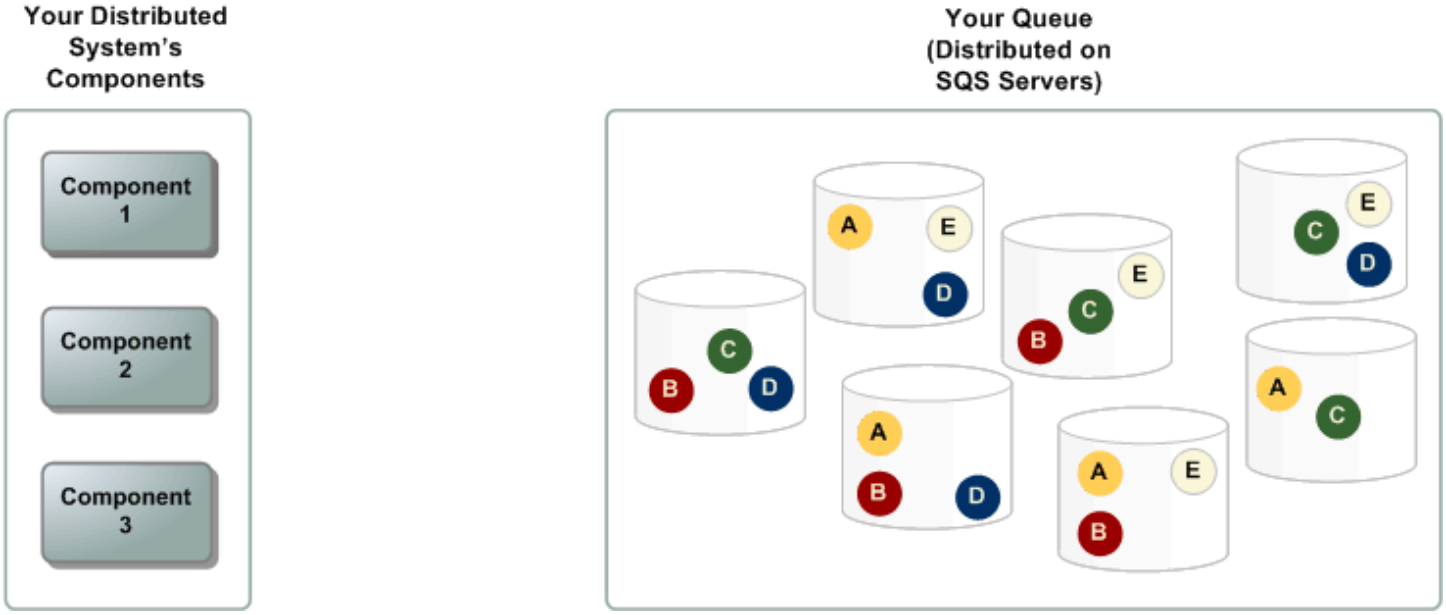
Architecture de base Amazon SQS

Cette section décrit les composants d'un système de messagerie distribué et explique le cycle de vie d'un message Amazon SQS.

Files d'attente distribuées

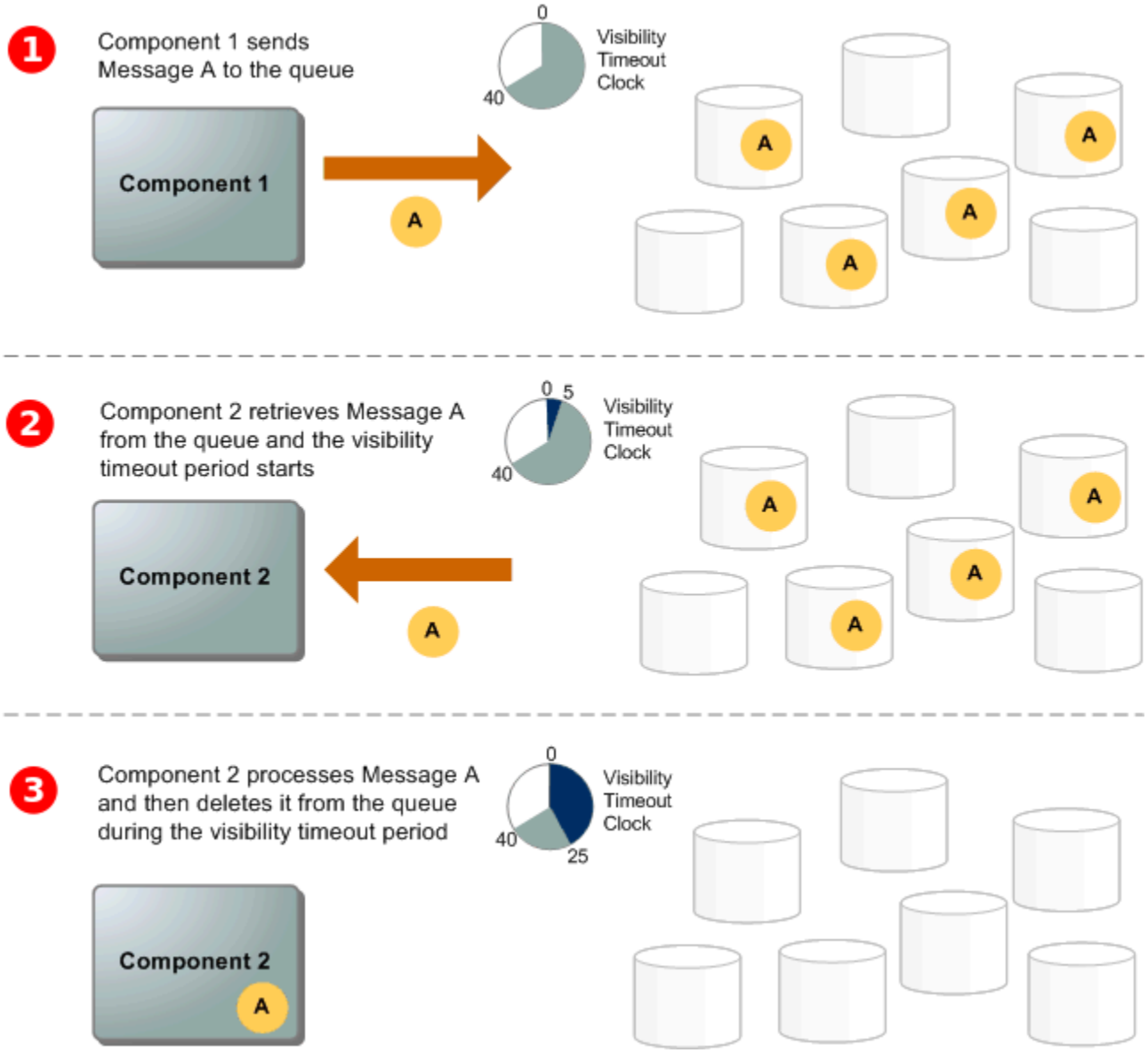
Un système de messagerie distribué comprend trois éléments principaux : les composants de votre système distribué, votre file d'attente (distribuée sur des serveurs Amazon SQS) et les messages de la file d'attente.

Dans le scénario suivant, le système comprend plusieurs producteurs (composants qui envoient des messages à la file d'attente) et plusieurs consommateurs (composants qui reçoivent des messages de la file d'attente). La file d'attente (qui contient les messages A à E) stocke les messages de manière redondante sur plusieurs serveurs Amazon SQS.



Cycle de vie des messages

Le scénario suivant décrit le cycle de vie d'un message Amazon SQS dans une file d'attente, de sa création à sa suppression.



1

Un producteur (composant 1) envoie un message A à une file d'attente, et le message est distribué de façon redondante entre les serveurs Amazon SQS.

2

Lorsqu'un consommateur (composant 2) est prêt à traiter des messages, il consomme les messages dans la file d'attente et le message A est renvoyé. Pendant son traitement, le message A reste dans

la file d'attente et n'est pas renvoyé aux demandes de réception suivantes pendant la durée du délai de visibilité.

3

Le consommateur (composant 2) supprime le message A de la file d'attente afin d'éviter que le message ne soit de nouveau reçu et traité à l'expiration du délai de visibilité.

> **Note**
>
> Amazon SQS supprime automatiquement d'une file d'attente les messages qui dépassent la période maximale de conservation des messages. La période de conservation des messages par défaut est de 4 jours. Cependant, vous pouvez configurer la période de rétention des messages sur une valeur allant de 60 secondes à 1 209 600 secondes (14 jours) avec `SetQueueAttributes`

Différences entre Amazon SQS, Amazon MQ et Amazon SNS

Amazon SQS, Amazon SNS et Amazon MQ proposent des services de messagerie gérés easy-to-use et hautement évolutifs, chacun étant conçu pour des rôles spécifiques au sein de systèmes distribués. Voici un aperçu détaillé des différences entre ces services :

Amazon SQS dissocie et fait évoluer les systèmes et composants logiciels distribués en tant que service de file d'attente. Il traite généralement les messages par l'intermédiaire d'un seul abonné, ce qui est idéal pour les flux de travail où la prévention des commandes et des pertes est essentielle. Pour une distribution plus large, l'intégration d'Amazon SQS à Amazon SNS permet d'utiliser un modèle de messagerie en fanout, transmettant efficacement les messages à plusieurs abonnés à la fois.

Amazon SNS permet aux éditeurs d'envoyer des messages à plusieurs abonnés par le biais de rubriques servant de canaux de communication. Les abonnés reçoivent les messages publiés via un type de point de terminaison compatible Amazon Data Firehose, tel qu'Amazon SQS, Lambda, HTTP, les e-mails, les notifications push mobiles et les SMS. Ce service est idéal pour les scénarios nécessitant des notifications immédiates, tels que l'engagement des utilisateurs en temps réel ou les systèmes d'alarme. Pour éviter la perte de messages lorsque les abonnés sont hors ligne, l'intégration d'Amazon SNS aux messages de file d'attente Amazon SQS garantit une diffusion cohérente.

Amazon MQ convient parfaitement aux entreprises qui souhaitent migrer depuis des courtiers de messages traditionnels, prenant en charge les protocoles de messagerie standard tels que AMQP et MQTT, ainsi qu'Apache ActiveMQ et RabbitMQ. Il est compatible avec les systèmes existants nécessitant une messagerie stable et fiable sans reconfiguration importante.

Le tableau suivant donne un aperçu du type de ressource de chaque service :

Type de ressource	Amazon SNS	Amazon SQS	Amazon MQ
Synchrone	Non	Non	Oui
asynchrone	Oui	Oui	Oui
Files d'attente	Non	Oui	Oui
Messagerie par publication-abonnement	Oui	Non	Oui
Agents de messages	Non	Non	Oui

Amazon SQS et Amazon SNS sont recommandés pour les nouvelles applications qui peuvent bénéficier d'une capacité de mise à l'échelle presque illimitée et d'API simples. Ils proposent généralement des solutions plus rentables pour les applications à volume élevé grâce à leurs pay-as-you-go prix. Nous recommandons Amazon MQ pour la migration d'applications provenant de courtiers de messages existants qui reposent sur la compatibilité avec des API telles que JMS ou des protocoles tels que le protocole AMQP (Advanced Message Queuing Protocol), le MQTT et le protocole STOMP (Simple Text Oriented Message Protocol). OpenWire

Configuration d'Amazon SQS

Avant de pouvoir utiliser Amazon SQS pour la première fois, vous devez suivre la procédure ci-dessous.

Rubriques

Étape 1 : créer un utilisateur Compte AWS et IAM

Pour accéder à un AWS service, vous devez d'abord créer un Compte AWScompte Amazon.com permettant d'utiliser AWS des produits. Vous pouvez utiliser votre Compte AWS pour consulter vos rapports d'activité et d'utilisation et pour gérer l'authentification et l'accès.

Pour éviter d'utiliser votre utilisateur Compte AWS root pour les actions Amazon SQS, il est recommandé de créer un utilisateur IAM pour chaque personne ayant besoin d'un accès administratif à Amazon SQS.

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.

Étape 2 : Accorder un accès par programmation

Pour utiliser les actions Amazon SQS (par exemple, en utilisant Java ou via le AWS Command Line Interface), vous avez besoin d'un ID de clé d'accès et d'une clé d'accès secrète.

> **Note**
>
> L'ID de clé d'accès et la clé d'accès secrète sont spécifiques à AWS Identity and Access Management. Ne les confondez pas avec les informations d'identification d'autres AWS services, tels que les paires de clés Amazon EC2.

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 demandes programmatiques adressées aux AWS SDK ou AWS aux API.	Suivez les instructions de l'interface que vous souhaitez utiliser. • Pour le AWS CLI, voir Authentification à l'aide des informations d'identification utilisateur IAM dans le guide

Quel utilisateur a besoin d'un accès programmatique ?	Pour	Par
		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.

Étape 3 : Préparation à l'utilisation de l'exemple de code

Ce guide contient des exemples d'utilisation du AWS SDK for Java. Pour exécuter l'exemple de code, suivez les instructions de configuration décrites dans Démarrer avec le kit AWS SDK pour Java 2.0.

Vous pouvez développer des AWS applications dans d'autres langages de programmation, tels que GoJavaScript, Python et Ruby. Pour plus d'informations, consultez la section Outils sur lesquels vous pouvez vous appuyer AWS.

> **Note**
>
> Vous pouvez explorer Amazon SQS sans écrire de code à l'aide d'outils tels que AWS Command Line Interface (AWS CLI) ou Windows. PowerShell Vous trouverez des AWS CLI exemples dans la section Amazon SQS de la AWS CLI Command Reference. Vous trouverez des PowerShell exemples de Windows dans la section Amazon Simple Queue Service de la référence des AWS Tools for PowerShell applets de commande.

Étapes suivantes

Vous pouvez maintenant commencer à gérer les files d'attente et les messages Amazon SQS à l'aide de l' AWS Management Console.

Démarrer avec Amazon SQS

Dans cette section, vous allez apprendre à créer des files d'attente standard ou FIFO à l'aide de la console Amazon SQS.

Rubriques

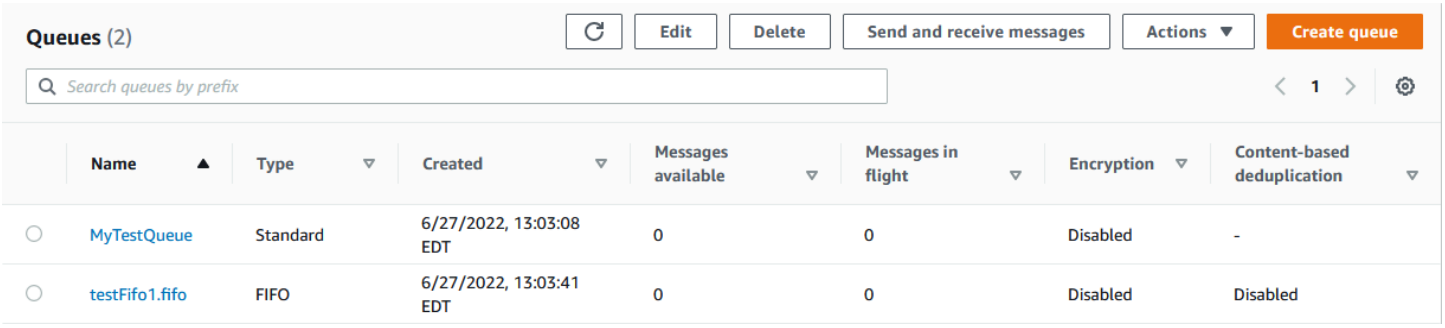
Prérequis

Avant de commencer, complétez les étapes détaillées dans Configuration d'Amazon SQS.

Comprendre la console Amazon SQS

Lorsque vous ouvrez la console Amazon SQS, choisissez Queues dans le volet de navigation. La page Files d'attente fournit des informations sur toutes vos files d'attente dans la région active.

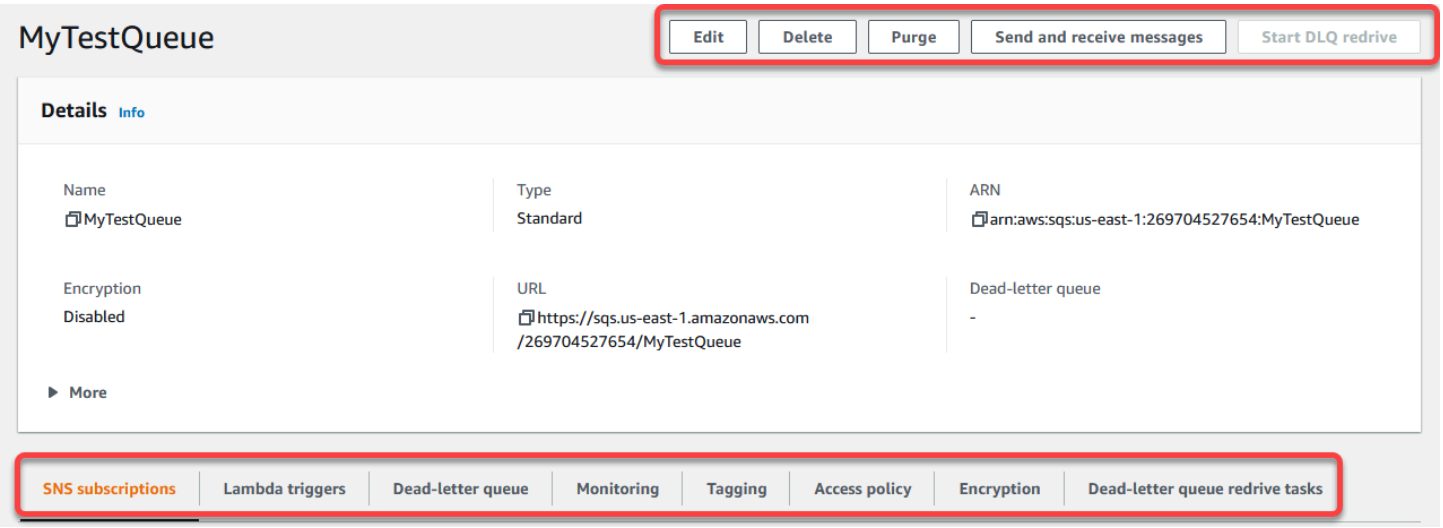
Chaque entrée de file d'attente fournit des informations essentielles sur la file d'attente, notamment son type et ses principaux attributs. Les files d'attente standard, optimisées pour un débit maximal et un classement optimal des messages, se distinguent des files d'attente du premier entré, premier sorti (FIFO), qui privilégient l'ordre des messages et leur caractère unique pour les applications nécessitant un séquençage strict des messages.



Éléments et actions interactifs

Sur la page Files d'attente, vous disposez de plusieurs options pour gérer vos files d'attente :

1. Actions rapides — À côté du nom de chaque file d'attente, un menu déroulant permet d'accéder rapidement aux actions courantes telles que l'envoi de messages, l'affichage ou la suppression de messages, la configuration de déclencheurs et la suppression de la file d'attente elle-même.
2. Vue détaillée et configuration : cliquez sur le nom d'une file d'attente pour ouvrir sa page de détails, dans laquelle vous pouvez approfondir les paramètres et les configurations des files d'attente. Ici, vous pouvez ajuster des paramètres tels que la période de rétention des messages, le délai de visibilité et la taille maximale des messages pour adapter la file d'attente aux exigences de votre application.



Sélection de régions et balises de ressources

Assurez-vous que vous êtes bien placé Région AWS pour accéder à vos files d'attente et les gérer efficacement. En outre, pensez à utiliser des balises de ressources pour organiser et classer vos files d'attente, afin d'améliorer la gestion des ressources, la répartition des coûts et le contrôle d'accès au sein de votre environnement AWS partagé.

En tirant parti des caractéristiques et fonctionnalités proposées dans la console Amazon SQS, vous pouvez gérer efficacement votre infrastructure de messagerie, optimiser les performances des files d'attente et garantir une distribution fiable des messages pour vos applications.

Types de files d'attente Amazon SQS

Amazon SQS prend en charge deux types de files d'attente : les files d'attente standard et les files d'attente FIFO. Utilisez les informations du tableau suivant pour choisir la file d'attente adaptée à votre situation. Pour en savoir plus sur les files d'attente Amazon SQS, consultez Commencer à utiliser les files d'attente standard Amazon SQS et Commencer à utiliser les files d'attente FIFO dans Amazon SQS.

Files d'attente standard	Files d'attente FIFO
Débit illimité : les files d'attente standard prennent en charge un nombre presque illimité d'appels d'API par seconde, par action d'API (`SendMessage` , `ReceiveMessage` ou `DeleteMessage`). Remise au moins une fois : un message est remis au moins une fois, mais il peut arriver qu'il soit remis en plusieurs exemplaires. Ordre dans la mesure du possible : il peut arriver que les messages soient remis dans un ordre différent de celui de leur envoi.	Débit élevé : si vous utilisez le traitement par lots, les files d'attente FIFO prennent en charge jusqu'à 3 000 messages par seconde, par méthode d'API (`SendMessageBatch` , `ReceiveMessage` ou `DeleteMessageBatch`). Les 3 000 messages par seconde représentent 300 appels d'API, chacun avec un lot de 10 messages. Pour demander une augmentation de quota, envoyez une demande de support. Sans traitement par lots, les files d'attente FIFO prennent en charge jusqu'à 300 appels d'API par seconde, par méthode d'API (`SendMessage` , `ReceiveMessage` ou `DeleteMessage`). Traitement en une seule fois : un message est remis une fois et reste disponible jusqu'à son traitement et sa suppression par un destinataire. Aucun doublon n'est ajouté à la file d'attente. Premier entré, premier sorti : l'ordre d'envoi et de réception des messages est rigoureusement conservé.

Files d'attente standard	Files d'attente FIFO
	
Envoyez les données entre les applications lorsque le débit est important, par exemple : • Découplez les demandes utilisateur en direct et le travail intensif en arrière-plan : permettez aux utilisateurs de charger un support pendant le redimensionnement ou le codage. • Allouez des tâches à plusieurs nœuds de travail : traitez un grand nombre de demandes de validations de cartes de crédit. • Organisez les messages en lots pour un traitement futur : planifiez l'ajout des entrées multiples dans une base de données.	Envoyez les données entre les applications lorsque l'ordre des événements est important, par exemple : • Veiller à ce que les commandes entrées par l'utilisateur soient exécutées dans l'ordre approprié. • Afficher le prix correct du produit en envoyant les modifications de prix dans le bon ordre. • Empêcher un étudiant de s'inscrire à un cours avant d'avoir créé un compte.

Création d'une file d'attente standard Amazon SQS et envoi d'un message

Voici comment créer une file d'attente standard pour Amazon SQS.

Création d'une file d'attente à l'aide de la console Amazon SQS

Vous pouvez utiliser la console Amazon SQS pour créer des files d'attente standard. La console fournit des valeurs par défaut pour tous les paramètres, à l'exception du nom de la file d'attente.

> **Important**
>
> Le 17 août 2022, le chiffrement côté serveur (SSE) par défaut a été appliqué à toutes les files d'attente Amazon SQS.

N'ajoutez pas de données d'identification personnelle (PII) ou d'autres données confidentielles ou sensibles dans les noms de file d'attente. Les noms des files d'attente sont accessibles à de nombreux Amazon Web Services, y compris les noms de facturation et CloudWatch les journaux. Les noms de file d'attente ne sont pas destinés à être utilisés pour des données privées ou sensibles.

Pour créer une file d'attente standard Amazon SQS

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
2. Choisissez Créez une file d'attente.
3. Pour Type, le type de file d'attente standard est défini par défaut.

   > **Note**
   >
   > Une fois la file d'attente créée, vous ne pouvez pas modifier son type.

4. Entrez un nom pour votre file d'attente.
5. (Facultatif) La console définit les valeurs par défaut pour les paramètres de configuration de la file d'attente. Sous Configuration, vous pouvez définir de nouvelles valeurs pour les paramètres suivants :

   a. Pour le Délai de visibilité, saisissez la durée et les unités. La plage est comprise entre 0 seconde et 12 heures. La valeur par défaut est de 30 secondes.

   b. Pour Période de conservation des messages, saisissez la durée et les unités. La plage est comprise entre 1 minute et 14 jours. La valeur par défaut est de 4 jours.

   c. Pour Retard de diffusion, saisissez la durée et les unités. La plage est comprise entre 0 seconde et 15 minutes. La valeur par défaut est de 0 seconde.

   d. Pour Taille maximale du message, saisissez une valeur. La plage est comprise entre 1 et 256 Ko. La valeur par défaut est de 256 Ko.

   e. Pour le Temps d'attente du message de réception, saisissez une valeur. La plage est comprise entre 0 et 20 secondes. La valeur par défaut est 0 seconde, qui permet de définir la recherche courte. Toute valeur différente de zéro définit une recherche longue.

6. (Facultatif) Définissez une stratégie d'accès. La stratégie d'accès définit les comptes, les utilisateurs et les rôles qui peuvent accéder à la file d'attente. La stratégie d'accès définit également les actions (telles que `SendMessage`, `ReceiveMessage` ou`DeleteMessage`)

auxquelles les utilisateurs peuvent accéder. La stratégie par défaut permet uniquement au propriétaire de la file d'attente d'envoyer et de recevoir des messages.

Pour définir la stratégie d'accès, effectuez l'une des opérations suivantes :

- Choisissez Basique pour configurer qui peut envoyer des messages à la file d'attente et qui peut recevoir des messages depuis la file d'attente. La console crée la stratégie en fonction de vos choix et affiche la stratégie d'accès qui en résulte dans le panneau JSON en lecture seule.
- Choisissez Avancé pour modifier directement la stratégie d'accès JSON. Cela vous permet de spécifier un ensemble personnalisé d'actions que chaque mandataire (compte, utilisateur ou rôle) peut effectuer.

7. Pour la Stratégie d'autorisation de redirection, choisissez Activé. Sélectionnez l'une des options suivantes : Tout autoriser (par défaut), Par file d'attente ou Refuser tout. Lorsque vous choisissez Par file d'attente, spécifiez une liste de 10 files d'attente source maximum en fonction de l'Amazon Resource Name (ARN).
8. Amazon SQS fournit un chiffrement côté serveur géré par défaut. Pour choisir un type de clé de chiffrement ou pour désactiver le chiffrement côté serveur géré par Amazon SQS, développez Chiffrement. Pour en savoir plus sur les types de clés de chiffrement, consultez Configuration du chiffrement côté serveur pour une file d'attente à l'aide de clés de chiffrement gérées par SQL et Configuration du chiffrement côté serveur pour une file d'attente à l'aide de la console Amazon SQS.

   > **Note**
   >
   > Lorsque SSE est activé, les demandes anonymes `SendMessage` et `ReceiveMessage` adressées à la file d'attente chiffrée sont rejetées. Les bonnes pratiques de sécurité d'Amazon SQS recommandent de ne pas utiliser de demandes anonymes. Si vous souhaitez envoyer des demandes anonymes à une file d'attente Amazon SQS, veillez à désactiver SSE.

9. (Facultatif) Pour configurer une file d'attente de lettres mortes pour recevoir des messages non distribuables, développez File d'attente de lettres mortes.
10. (Facultatif) Pour ajouter des balises à la file d'attente, développez Balises.
11. Choisissez Créez une file d'attente. Amazon SQS crée la file d'attente et affiche la page de Détails de la file d'attente.

Amazon SQS diffuse les informations relatives à la nouvelle file d'attente dans le système. Amazon SQS étant un système distribué, il se peut que la console affiche la file d'attente sur la page Files d'attente avec un léger retard.

Envoyer un message

Après avoir créé votre file d'attente, vous pouvez lui envoyer un message.

1. Dans le volet de navigation de gauche, choisissez Files d'attente. Dans la liste des files d'attente, sélectionnez la file d'attente que vous avez créée.
2. Dans Actions, choisissez Envoyer et recevoir des messages.

   La console affiche la page Envoyer et recevoir des messages.
3. Dans le Corps du message, saisissez le texte du message.
4. Pour une file d'attente standard, vous pouvez saisir une valeur pour le Délai de livraison et choisir les unités. Par exemple, saisissez `60` et choisissez secondes. Pour plus d'informations, consultez Temporisateurs de messages Amazon SQS.
5. Choisissez Send Message (Envoyer un message).

   Lorsque votre message est envoyé, la console affiche un message de réussite. Choisissez Afficher les détails pour afficher les informations relatives au message envoyé.

Création d'une file d'attente FIFO Amazon SQS et envoi d'un message

Voici comment créer une file d'attente FIFO pour Amazon SQS.

Créer une file d'attente

Vous pouvez utiliser la console Amazon SQS pour créer des files d'attente FIFO. La console fournit des valeurs par défaut pour tous les paramètres, à l'exception du nom de la file d'attente.

> **Important**
>
> Le 17 août 2022, le chiffrement côté serveur (SSE) par défaut a été appliqué à toutes les files d'attente Amazon SQS.
> N'ajoutez pas de données d'identification personnelle (PII) ou d'autres données confidentielles ou sensibles dans les noms de file d'attente. Les noms des files d'attente sont

accessibles à de nombreux Amazon Web Services, y compris les noms de facturation et CloudWatch les journaux. Les noms de file d'attente ne sont pas destinés à être utilisés pour des données privées ou sensibles.

Pour créer une file d'attente FIFO Amazon SQS

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
2. Choisissez Créez une file d'attente.
3. Pour Type, le type de file d'attente standard est défini par défaut. Pour créer une file d'attente FIFO, choisissez FIFO.

   > **Note**
   >
   > Une fois la file d'attente créée, vous ne pouvez pas modifier son type.

4. Entrez un nom pour votre file d'attente.

   Le nom d'une file d'attente FIFO doit se terminer par le suffixe `.fifo`. Le suffixe est pris en compte dans le quota de 80 caractères pour les noms de file d'attente. Pour déterminer si une file d'attente est de type FIFO, vous pouvez vérifier si son nom se termine par le suffixe.

5. (Facultatif) La console définit les valeurs par défaut pour les paramètres de configuration de la file d'attente. Sous Configuration, vous pouvez définir de nouvelles valeurs pour les paramètres suivants :

   a. Pour le Délai de visibilité, saisissez la durée et les unités. La plage est comprise entre 0 seconde et 12 heures. La valeur par défaut est de 30 secondes.

   b. Pour Période de conservation des messages, saisissez la durée et les unités. La plage est comprise entre 1 minute et 14 jours. La valeur par défaut est de 4 jours.

   c. Pour Retard de diffusion, saisissez la durée et les unités. La plage est comprise entre 0 seconde et 15 minutes. La valeur par défaut est de 0 seconde.

   d. Pour Taille maximale du message, saisissez une valeur. La plage est comprise entre 1 et 256 Ko. La valeur par défaut est de 256 Ko.

   e. Pour le Temps d'attente du message de réception, saisissez une valeur. La plage est comprise entre 0 et 20 secondes. La valeur par défaut est 0 seconde, qui permet de définir la recherche courte. Toute valeur différente de zéro définit une recherche longue.

f. Pour une file d'attente FIFO, choisissez Déduplication basée sur le contenu pour activer cette option. Par défaut, ce paramètre est désactivé.

g. (Facultatif) Pour qu'une file d'attente FIFO permette un débit plus élevé pour l'envoi et la réception de messages dans la file d'attente, choisissez Activer le FIFO à haut débit.

   Le choix de cette option modifie les options associées (Portée de la déduplication et Limite de débit FIFO) en fonction des paramètres requis pour activer un débit élevé pour les files d'attente FIFO. Si vous modifiez l'un des paramètres requis pour utiliser le FIFO à débit élevé, le débit normal est effectif pour la file d'attente et la déduplication se produit comme indiqué. Pour plus d'informations, consultez Débit élevé pour les files d'attente FIFO dans Amazon SQS et Quotas de messages Amazon SQS.

6. (Facultatif) Définissez une stratégie d'accès. La stratégie d'accès définit les comptes, les utilisateurs et les rôles qui peuvent accéder à la file d'attente. La stratégie d'accès définit également les actions (telles que `SendMessage`, `ReceiveMessage` ou`DeleteMessage`) auxquelles les utilisateurs peuvent accéder. La stratégie par défaut permet uniquement au propriétaire de la file d'attente d'envoyer et de recevoir des messages.

   Pour définir la stratégie d'accès, effectuez l'une des opérations suivantes :

   - Choisissez Basique pour configurer qui peut envoyer des messages à la file d'attente et qui peut recevoir des messages depuis la file d'attente. La console crée la stratégie en fonction de vos choix et affiche la stratégie d'accès qui en résulte dans le panneau JSON en lecture seule.
   - Choisissez Avancé pour modifier directement la stratégie d'accès JSON. Cela vous permet de spécifier un ensemble personnalisé d'actions que chaque mandataire (compte, utilisateur ou rôle) peut effectuer.

7. Pour la Stratégie d'autorisation de redirection, choisissez Activé. Sélectionnez l'une des options suivantes : Tout autoriser (par défaut), Par file d'attente ou Refuser tout. Lorsque vous choisissez Par file d'attente, spécifiez une liste de 10 files d'attente source maximum en fonction de l'Amazon Resource Name (ARN).

8. Amazon SQS fournit un chiffrement côté serveur géré par défaut. Pour choisir un type de clé de chiffrement ou pour désactiver le chiffrement côté serveur géré par Amazon SQS, développez Chiffrement. Pour en savoir plus sur les types de clés de chiffrement, consultez Configuration du chiffrement côté serveur pour une file d'attente à l'aide de clés de chiffrement gérées par SQL et Configuration du chiffrement côté serveur pour une file d'attente à l'aide de la console Amazon SQS.

> **Note**
>
> Lorsque SSE est activé, les demandes anonymes `SendMessage` et `ReceiveMessage` adressées à la file d'attente chiffrée sont rejetées. Les bonnes pratiques de sécurité d'Amazon SQS recommandent de ne pas utiliser de demandes anonymes. Si vous souhaitez envoyer des demandes anonymes à une file d'attente Amazon SQS, veillez à désactiver SSE.

9. (Facultatif) Pour configurer une file d'attente de lettres mortes pour recevoir des messages non distribuables, développez File d'attente de lettres mortes.
10. (Facultatif) Pour ajouter des balises à la file d'attente, développez Balises.
11. Choisissez Créez une file d'attente. Amazon SQS crée la file d'attente et affiche la page de Détails de la file d'attente.

Amazon SQS diffuse les informations relatives à la nouvelle file d'attente dans le système. Amazon SQS étant un système distribué, il se peut que la console affiche la file d'attente sur la page Files d'attente avec un léger retard.

Après avoir créé une file d'attente, vous pouvez lui envoyer des messages, et recevoir et supprimer des messages. Vous pouvez également modifier tous les paramètres de configuration de la file d'attente, à l'exception du type de file d'attente.

Envoyer un message

Après avoir créé votre file d'attente, vous pouvez lui envoyer un message.

1. Dans le volet de navigation de gauche, choisissez Files d'attente. Dans la liste des files d'attente, sélectionnez la file d'attente que vous avez créée.
2. Dans Actions, choisissez Envoyer et recevoir des messages.

   La console affiche la page Envoyer et recevoir des messages.
3. Dans le Corps du message, saisissez le texte du message.
4. Pour une file d'attente FIFO (First-In First-Out), entrez un ID de groupe de messages. Pour plus d’informations, consultez Logique de distribution des files d'attente FIFO dans Amazon SQS.
5. (Facultatif) Pour une file d'attente FIFO, vous pouvez saisir un ID de déduplication des messages. Si vous avez activé la déduplication basée sur le contenu pour la file d'attente, l'ID

de déduplication des messages n'est pas requis. Pour plus d’informations, consultez Logique de distribution des files d'attente FIFO dans Amazon SQS.

6. Les files d'attente FIFO ne prennent pas en charge les temporisateurs pour les messages individuels. Pour plus d’informations, consultez Temporisateurs de messages Amazon SQS.
7. Choisissez Send Message (Envoyer un message).

   Lorsque votre message est envoyé, la console affiche un message de réussite. Choisissez Afficher les détails pour afficher les informations relatives au message envoyé.

Gérer une file d'attente Amazon SQS

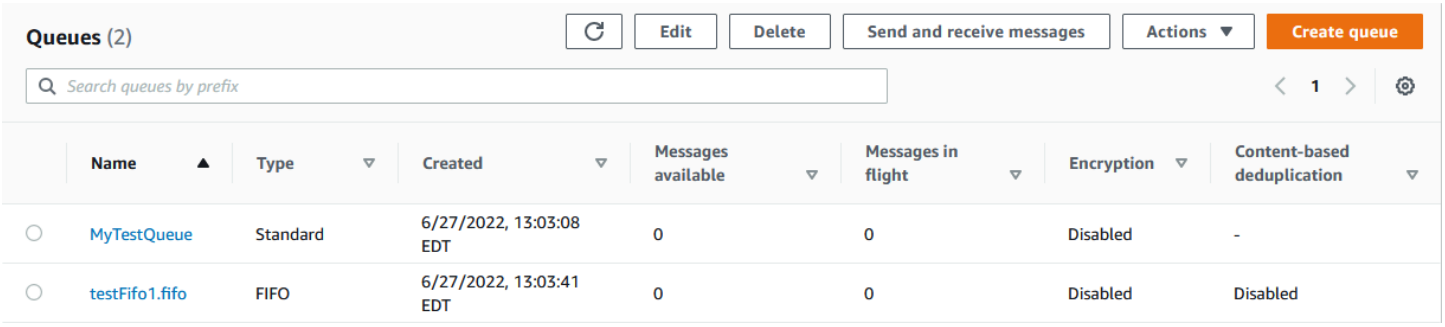
Cette section vous aide à vous familiariser avec Amazon SQS en vous montrant comment gérer les files d'attente et les messages avec la console Amazon SQS.

Prérequis

Avant de commencer, complétez les étapes détaillées dans Configuration d'Amazon SQS.

Comprendre la console Amazon SQS

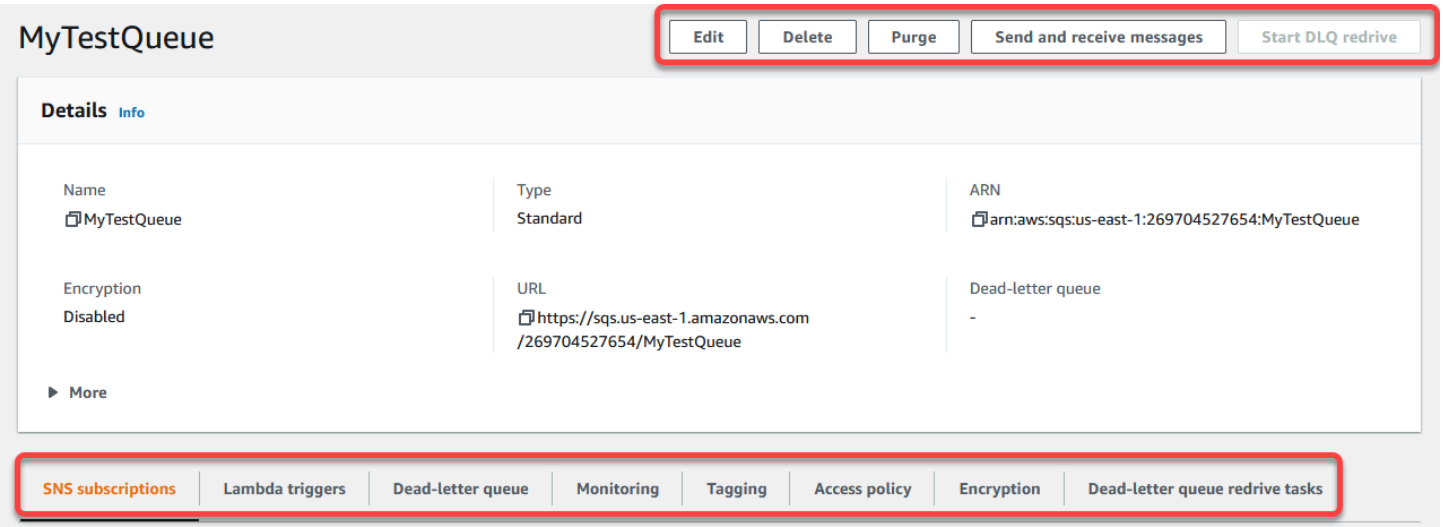
Lorsque vous ouvrez la console, choisissez Files d'attente dans le volet de navigation pour afficher la page Files d'attente. La page Files d'attente fournit des informations sur toutes vos files d'attente dans la région active.



L'entrée de chaque file d'attente indique le type de file d'attente et d'autres informations la concernant. La colonne Type vous permet de distinguer en un coup d'œil les files d'attente standard des files d'attente FIFO (First-In First-Out).

Sur la page Files d'attente, vous pouvez effectuer des actions sur une file d'attente de deux manières. Vous pouvez choisir l'option à côté du nom de la file d'attente, puis choisir l'action que vous souhaitez effectuer sur la file d'attente.

Vous pouvez également choisir le nom de la file d'attente, ce qui ouvre la page Détails de cette file d'attente. La page Détails inclut les mêmes actions que la page Files d'attente. En outre, vous pouvez choisir l'un des onglets situés sous la section Détails pour afficher des informations de configuration et des actions supplémentaires.



Modification d'une file d'attente Amazon SQS à l'aide de la console

Vous pouvez utiliser la console Amazon SQS pour modifier tous les paramètres de configuration de file d'attente (à l'exception du type de file d'attente) et ajouter ou supprimer des fonctionnalités de file d'attente.

Pour modifier une file d'attente Amazon SQS (console)

1. Ouvrez la page Files d'attente de la console Amazon SQS.
2. Sélectionnez une file d'attente, puis choisissez Modifier.
3. (Facultatif) Sous Configuration, mettez à jour les paramètres de configuration de la file d'attente.
4. (Facultatif) Pour mettre à jour la stratégie d'accès, sous Stratégie d'accès, modifiez la stratégie JSON.
5. (Facultatif) Pour mettre à jour une stratégie d'autorisation de redirection de file d'attente de lettres mortes, développez Stratégie d'autorisation de redirection.
6. (Facultatif) Pour mettre à jour ou supprimer le chiffrement, développez Chiffrement.
7. (Facultatif) Pour ajouter, mettre à jour ou supprimer une file d'attente de lettres mortes (qui vous permet de recevoir des messages non distribuables), développez File d'attente de lettres mortes.
8. (Facultatif) Pour ajouter, mettre à jour ou supprimer les balises de la file d'attente, développez Balises.
9. Choisissez Enregistrer.

   La console affiche la page Détails de la file d'attente.

Réception et suppression d'un message dans Amazon SQS

Après avoir envoyé des messages à une file d'attente Amazon SQS, vous avez la possibilité de les recevoir et de les supprimer. Lorsque vous demandez des messages à partir d'une file d'attente, vous ne pouvez pas spécifier de messages individuels. Vous déterminez plutôt le nombre maximum de messages que vous souhaitez récupérer, dans la limite de 10.

Amazon SQS fonctionne comme un système distribué, ce qui peut parfois entraîner une réponse vide lors de la récupération de messages dans une file d'attente contenant peu de messages. Dans ce cas, il vous suffit de réexécuter votre demande. Pour optimiser la récupération des messages et minimiser les réponses vides, pensez à utiliser des interrogations longues. Un long sondage retarde la réponse jusqu'à ce qu'un message soit disponible ou que le sondage expire, ce qui réduit les coûts de sondage inutiles et améliore l'efficacité.

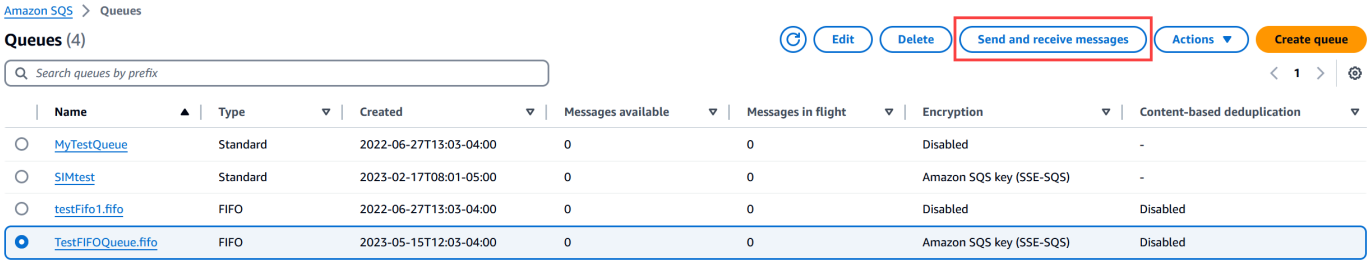
Les messages ne sont pas automatiquement supprimés après leur extraction, car Amazon SQS garantit que vous ne perdez pas l'accès à un message en raison d'échecs de traitement, tels que des problèmes liés à votre application ou des perturbations du réseau. Pour supprimer définitivement un message de la file d'attente, vous devez explicitement envoyer une demande de suppression après le traitement du message afin de confirmer la réception et le traitement réussis.

Lorsque les messages sont récupérés via la console Amazon SQS, ils sont immédiatement revisibles pour être récupérés à nouveau. Ce comportement par défaut garantit que les messages ne sont pas perdus par inadvertance lors d'opérations manuelles, mais qu'ils peuvent entraîner des traitements répétés. Dans les environnements automatisés, ajustez le paramètre de délai de visibilité pour contrôler la durée pendant laquelle un message reste invisible pour les autres consommateurs après avoir été récupéré. Ce paramètre est essentiel pour coordonner le traitement des messages entre plusieurs consommateurs et garantir que les messages ne sont traités qu'une seule fois.

Pour des opérations plus détaillées sur la réception et la suppression de messages, consultez le guide de référence des API Amazon SQS. Ce guide fournit des informations complètes sur les points de terminaison des API, y compris les paramètres permettant de gérer efficacement les scénarios complexes de gestion des messages.

Pour recevoir et supprimer un message à l'aide de la console

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
2. Dans le volet de navigation, choisissez Files d'attente.
3. Sur la page Files d'attente, sélectionnez une file d'attente, puis choisissez Envoyer et recevoir des messages.



4. Sur la page Envoyer et recevoir des messages, choisissez Sondage pour les messages.

   Amazon SQS commence à rechercher les messages présents dans la file d'attente. La barre de progression située sur le côté droit de la section Recevoir des messages affiche la durée de la recherche.

   La section Messages affiche la liste des messages reçus. Pour chaque message, la liste affiche l'ID du message, la date d'envoi, la taille et le nombre de destinataires.
5. Pour supprimer des messages, choisissez les messages que vous souhaitez supprimer, puis sélectionnez Supprimer.
6. Dans la boîte de dialogue Supprimer les messages, choisissez Supprimer.

Confirmation qu'une file d'attente Amazon SQS est vide

Dans la plupart des cas, vous pouvez utiliser une recherche prolongée pour déterminer si une file d'attente est vide. Dans de rares cas, vous pouvez recevoir des réponses vides même si une file d'attente contient encore des messages, en particulier si vous avez spécifié une faible valeur pour le paramètre Temps d'attente du message de réception lorsque vous avez créé la file d'attente. Cette section explique comment vérifier qu'une file d'attente est vide.

Confirmer qu'une file d'attente est vide (console)

1. Empêchez tous les producteurs d'envoyer des messages.
2. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
3. Dans le volet de navigation, choisissez Files d'attente.
4. Sur la page Files d'attente, choisissez une file d'attente.
5. Sélectionnez l'onglet Monitoring (Surveillance).
6. En haut à droite des tableaux de bord de surveillance, cliquez sur la flèche vers le bas à côté du symbole Actualiser. Dans le menu déroulant, choisissez Actualisation automatique. Laissez l'intervalle d'actualisation sur 1 minute.

7. Observez les tableaux de bord suivants :

   - Nombre approximatif de messages retardés
   - Nombre approximatif de messages non visibles
   - Nombre approximatif de messages visibles

   Lorsque tous affichent des valeurs de `0` pendant plusieurs minutes, cela signifie que la file d'attente est vide.

Pour confirmer qu'une file d'attente est vide (AWS CLI, AWS API)

1. Empêchez tous les producteurs d'envoyer des messages.
2. Exécutez l'une des commandes suivantes à plusieurs reprises :

   - AWS CLI: `get-queue-attributes`
   - AWS API : `GetQueueAttributes`
3. Observez les métriques pour les attributs suivants :

   - `ApproximateNumberOfMessagesDelayed`
   - `ApproximateNumberOfMessagesNotVisible`
   - `ApproximateNumberOfMessagesVisible`

   Lorsque tous affichent la valeur `0` pendant plusieurs minutes, cela signifie que la file d'attente est vide.

Si vous vous fiez aux CloudWatch statistiques d'Amazon, assurez-vous de voir plusieurs points de données nuls consécutifs avant de considérer que cette file d'attente est vide. Pour plus d'informations sur CloudWatch les métriques, consultez CloudWatch Métriques disponibles pour Amazon SQS.

Supprimer une file d'attente Amazon SQS

Si vous n'utilisez plus une file d'attente Amazon SQS et que vous ne prévoyez pas de l'utiliser dans un futur proche, nous vous recommandons de la supprimer.

> **Tip**
>
> Si vous souhaitez vérifier qu'une file d'attente est vide avant de la supprimer, consultez Confirmation qu'une file d'attente Amazon SQS est vide.

Vous pouvez supprimer une file d'attente, même lorsqu'elle n'est pas vide. Pour supprimer les messages d'une file d'attente, mais pas la file d'attente elle-même, purgez la file d'attente.

Pour supprimer une file d'attente (console)

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
2. Dans le volet de navigation, choisissez Files d'attente.
3. Sur la page Files d'attente, choisissez la file d'attente à supprimer.
4. Sélectionnez Delete (Supprimer).
5. Dans la boîte de dialogue Supprimer la file d'attente, confirmez la suppression en saisissant **`delete`**.
6. Sélectionnez Delete (Supprimer).

Pour supprimer une file d'attente (AWS CLI et une API)

Vous pouvez utiliser l'une des commandes suivantes pour supprimer une file d'attente :

- AWS CLI: `aws sqs delete-queue`
- AWS API : `DeleteQueue`

Purger les messages d'une file d'attente à l'aide de la console Amazon SQS

Si vous ne voulez pas supprimer une file d'attente Amazon SQS, mais que vous devez supprimer tous les messages de celle-ci, vous pouvez purger la file d'attente. Le processus de suppression des messages peut prendre jusqu'à 60 secondes. Nous vous recommandons d'attendre 60 secondes, quelle que soit la taille de votre file d'attente.

> **Important**
>
> Lorsque vous purgez une file d'attente, vous ne pouvez récupérer aucun des messages supprimés.

Pour purger une file d'attente (console)

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
2. Dans le volet de navigation, choisissez Files d'attente.
3. Sur la page Files d'attente, choisissez la file d'attente à purger.
4. Dans Actions, choisissez Purger.
5. Dans la boîte de dialogue Purger la file d'attente, confirmez la purge en saisissant **`purge`** et en choisissant Purger.

   Tous les messages sont purgés de la file d'attente. La console affiche une bannière de confirmation.

Tâches courantes pour démarrer avec Amazon SQS

Maintenant que vous avez créé une file d'attente, et appris à envoyer, recevoir et supprimer des messages, et à supprimer une file d'attente, vous souhaiterez peut-être essayer les opérations suivantes :

- Pour déclencher une fonction Lambda, consultez Configuration d'une file d'attente Amazon SQS pour déclencher une fonction AWS Lambda.
- Découvrez comment configurer les files d'attente, notamment le SSE et d'autres fonctionnalités.
- Découvrez comment envoyer un message avec des attributs.
- Découvrez comment envoyer un message depuis un VPC.
- Pour en savoir plus sur les fonctionnalités et l'architecture d'Amazon SQS, consultez Types de files d'attente Amazon SQS et Architecture de base Amazon SQS.
- Pour obtenir des conseils et des mises en garde qui vous aideront à tirer le meilleur d'Amazon SQS, consultez Bonnes pratiques relatives à Amazon SQS.
- Explorez les exemples Amazon SQS relatifs à l'un des AWS SDK, tels que le guide du AWS SDK for Java 2.x développeur.

- Pour en savoir plus sur les AWS CLI commandes Amazon SQS, consultez la référence des AWS CLI commandes.
- Pour en savoir plus sur les actions Amazon SQS, consultez la Référence d'API Amazon Simple Queue Service.
- Découvrez comment interagir avec Amazon SQS par programmation : lisez Working with APIs et explorez le centre de développement :AWS
  - Java
  - JavaScript
  - PHP
  - Python
  - Ruby
  - Windows & .NET

- Découvrez comment surveiller les coûts et les ressources dans la section Résolution des problèmes dans Amazon SQS.
- Découvrez comment protéger vos données et y accéder dans la section Sécurité.
- Pour en savoir plus sur le flux de travail Amazon SQS, consultez la section sur le flux de travail du processus de contrôle d'accès Amazon SQS.

Commencer à utiliser les files d'attente standard Amazon SQS

Dans Amazon SQS, le type de file d'attente par défaut est dénommé standard. Les files d'attente standard prennent en charge un nombre presque illimité d'appels d'API par seconde, par action d'API (`SendMessage`, `ReceiveMessage` ou `DeleteMessage`). Les files d'attente standard prennent en charge la livraison des at-least-once messages. Cependant, il peut arriver (en raison de l'architecture hautement distribuée qui permet un débit presque illimité) que plusieurs copies d'un message soient diffusées dans le désordre. Les files d'attente standard classent au mieux les messages, ce qui signifie qu'ils sont généralement remis dans l'ordre de leur envoi.

Amazon SQS stocke de manière redondante un message dans plusieurs zones de disponibilité (AZ) avant qu'un message `SendMessage` ne soit reconnu. Les copies des messages étant stockées dans plusieurs zones de disponibilité, aucune défaillance d'ordinateur, de réseau ou de zone de disponibilité ne peut rendre les messages inaccessibles.

Pour plus d'informations sur la création et la configuration de files d'attente à l'aide de la console Amazon SQS, consultez Création d'une file d'attente à l'aide de la console Amazon SQS. Pour obtenir des exemples Java, consultez Exemples de SDK Java Amazon SQS.

Vous pouvez utiliser des files d'attente standard dans de nombreux scénarios, tant que votre application peut traiter des messages qui arrivent plusieurs fois et dans le désordre, par exemple :

- Découpler les demandes utilisateur en direct et le travail intensif en arrière-plan : permettez aux utilisateurs de charger un support pendant le redimensionnement ou le codage.
- Allouer des tâches à plusieurs composants master : traitez un grand nombre de demandes de validation de cartes de crédit.
- Organiser les messages en lots pour un traitement futur : planifiez l'ajout d'entrées multiples dans une base de données.

Pour connaître les quotas liés aux files d'attente standard, consultez Quotas.

Pour connaître les bonnes pratiques d'utilisation des files d'attente standard, consultez Recommandations pour les files d'attente Amazon SQS standard et FIFO.

Ordre des messages

Une file d'attente standard permet de conserver au mieux l'ordre des messages, mais plusieurs copies d'un message peuvent être remises dans le désordre. Si le système exige que l'ordre soit préservé, nous vous recommandons d'utiliser une file d'attente FIFO (First-In First-Out) ou d'ajouter des informations de séquencement dans chaque message afin que vous puissiez réorganiser les messages lors de leur réception.

Une t-least-once livraison

Amazon SQS stocke des copies de vos messages sur plusieurs serveurs à des fins de redondance et de haute disponibilité. Dans de rares occasions, l'un des serveurs qui stockent la copie d'un message peut être indisponible lors de la réception ou de la suppression d'un message.

Dans ce cas, la copie du message n'est pas supprimée sur le serveur qui n'est pas disponible et il est possible que vous obteniez à nouveau cette copie lorsque vous recevrez des messages. Concevez les applications afin qu'elles soient idempotentes (c.-à-d. qu'elles ne doivent pas être affectées si le même message est traité plus d'une fois).

Identifiants de files d'attente et de messages Amazon SQS

Cette section décrit les identifiants des files d'attentes standard et FIFO. Ces identifiants peuvent vous aider à trouver et à manipuler des files d'attente et des messages spécifiques.

Identifiants pour les files d'attente Amazon SQS standard

Pour plus d'informations sur les identifiants suivants, consultez la Référence d'API Amazon Simple Queue Service.

Nom et URL de la file d'attente

Lorsque vous créez une file d'attente, vous devez indiquer un nom unique pour le compte et la région AWS . Amazon SQS attribue à chaque file d'attente que vous créez un identifiant appelé URL de file d'attente qui inclut le nom de la file d'attente et les autres composants Amazon SQS. Chaque fois que vous souhaitez effectuer une action au niveau d'une file d'attente, vous devez fournir cette URL.

L'URL suivante est celle d'une file d'attente nommée `MyQueue`, qui appartient à un utilisateur dont le numéro de compte AWS est `123456789012`.

```
https://sqs.us-east-2.amazonaws.com/123456789012/MyQueue
```

Vous pouvez extraire l'URL d'une file d'attente par programmation en listant vos files d'attente et en analysant la chaîne qui suit le numéro de compte. Pour plus d'informations, consultez `ListQueues`.

ID de message

Chaque message reçoit un ID de message attribué par le système qu'Amazon SQS vous renvoie dans la réponse `SendMessage`. Cet identifiant est utile pour identifier les messages. La longueur maximale d'un ID de message est de 100 caractères.

Descripteur de réception

Chaque fois que vous recevez un message d'une file d'attente, vous recevez un descripteur de réception correspondant. Cette gestion est associée à la réception du message, et non au message lui-même. Pour supprimer le message ou pour en modifier la visibilité, vous devez fournir le descripteur de réception (et non l'ID du message). C'est pourquoi vous devez toujours recevoir un message avant de pouvoir le supprimer (vous ne pouvez pas placer un message dans la file d'attente, puis le rappeler). La longueur maximale d'un descripteur de réception est de 1 024 caractères.

> **Important**
>
> Si vous recevez un message plusieurs fois, chaque fois que vous le recevez, vous obtenez un descripteur de réception différent. Lorsque vous demandez la suppression du message, vous devez fournir le descripteur de réception le plus récent. Dans le cas contraire, la suppression peut ne pas fonctionner.

Voici un exemple de descripteur de réception (réparti sur trois lignes).

```
MbZj6wDWli+JvwwJaBV+3dcjk2YW2vA3+STFFljTM8tJJg6HRG6PYSasuWXPJB+Cw
Lj1FjgXUv1uSj1gUPAWV66FU/WeR4mq2OKpEGYWbnLmpRCJVAyeMjeU5ZBdtcQ+QE
auMZc8ZRv37sIW2iJKq3M9MFx1YvV11A2x/KSbkJ0=
```

Quotas

Le tableau suivant répertorie les quotas relatifs aux files d'attente standard.

Quota	Description
File d'attente à retardement	Le délai (minimum) par défaut pour une file d'attente est de 0 seconde. La valeur maximale est de 15 minutes.
Files d'attente répertoriées	1000 files d'attente par demande `ListQueues` .
Durée d'attente pour interrogation longue	Le temps d'attente maximal pour la recherche prolongée est de 20 secondes.
Messages par file d'attente (en attente)	Le nombre de messages qu'une file d'attente Amazon SQS peut stocker est illimité.
Messages par file d'attente (en transit)	Pour la plupart des files d'attente standard (en fonction du trafic de la file d'attente et du backlog de messages), il peut y avoir un maximum d'environ 120 000 messages en cours (reçus depuis une file d'attente par un consommateur, mais pas encore supprimés de la file d'attente). Si vous atteignez ce quota tout en utilisant la recherche courte, Amazon SQS renvoie le message d'erreur `OverLimit` . Si vous utilisez la recherche prolongée, Amazon SQS ne renvoie aucun message d'erreur. Pour éviter d'atteindre cette limite, supprimez les messages de la file d'attente une fois qu'ils ont été traités. Vous pouvez également augmenter le nombre de files d'attente que vous utilisez pour traiter vos messages. Pour demander une augmentation de quota, envoyez une demande de support.
Nom de la file d'attente	Le nom des files d'attente peut contenir jusqu'à 80 caractères. Les caractères suivants sont acceptés : caractères alphanumériques, tirets (-) et traits de soulignement (_).

Quota	Description
	Note Les noms de file d'attente sont sensible à la casse (par exemple, `Test-queue` et `test-queue` désignent des files d'attente différentes).
Balise de file d'attente	Nous vous déconseillons d'ajouter plus de 50 balises à une file d'attente. Le balisage prend en charge les caractères Unicode en UTF-8.
	La balise `Key` est obligatoire, mais la balise `Value` est facultative.
	La balise `Key` et la balise `Value` sont sensibles à la casse.
	La balise `Key` et la balise `Value` peuvent inclure des caractères alphanumériques Unicode en UTF-8 et des espaces blancs. Les caractères spéciaux suivants sont acceptés : `_ . : / = + - @`
	La balise `Key` ou `Value` ne doit pas inclure le préfixe réservé `aws:` (vous ne pouvez pas supprimer les clés de balise ou les valeurs ayant ce préfixe).
	La longueur `Key` de balise maximale est de 128 caractère s Unicode en UTF-8. La balise `Key` ne doit pas être vide ou null.
	La longueur `Value` de balise maximale est de 256 caractères Unicode en UTF-8. La balise `Value` peut être vide ou null.
	Les actions de balisage sont limitées à 30 TPS par action. Compte AWS Si votre application nécessite un débit plus élevé, soumettez une demande.

Commencer à utiliser les files d'attente FIFO dans Amazon SQS

En plus d'avoir toutes les fonctionnalités des files d'attente standard, les files d'attente FIFO (First-In First-Out) sont conçues pour la messagerie entre les applications lorsque l'ordre des opérations et des événements est essentiel, ou lorsque des doublons ne peuvent pas être tolérés.

Voici des exemples de situations dans lesquelles vous pourriez utiliser des files d'attente FIFO :

1. Système de gestion des commandes pour le commerce électronique où l'ordre est essentiel.
2. Intégration à un système tiers où les événements doivent être traités dans l'ordre
3. Traitement des entrées saisies par l'utilisateur dans l'ordre saisi
4. Communications et mise en réseau : envoi et réception de données et d'informations dans le même ordre
5. Systèmes informatiques : s'assurer que les commandes saisies par l'utilisateur sont exécutées dans le bon ordre
6. Établissements d'enseignement : empêcher un étudiant de s'inscrire à un cours avant d'avoir créé un compte
7. Système de billetterie en ligne : où les billets sont distribués selon le principe du premier arrivé, premier servi

> **Note**
>
> Les files d'attente FIFO fournissent également un traitement en une seule fois, mais avec un nombre limité de transactions par seconde (TPS). Vous pouvez utiliser le mode débit élevé d'Amazon SQS avec votre file d'attente FIFO pour augmenter votre limite de transactions. Pour plus de détails sur l'utilisation du mode débit élevé, consultez Débit élevé pour les files d'attente FIFO dans Amazon SQS. Pour plus d'informations sur les quotas de débit, consultez the section called “Quotas de messages”.

Les files d'attente FIFO Amazon SQS sont disponibles dans toutes les régions où Amazon SQS est disponible.

Pour en savoir plus sur l'utilisation des files d'attente FIFO pour les commandes complexes, consultez Résoudre les problèmes de commande complexes avec les files d'attente FIFO Amazon SQS.

Pour plus d'informations sur la création et la configuration de files d'attente à l'aide de la console Amazon SQS, consultez Création d'une file d'attente à l'aide de la console Amazon SQS. Pour obtenir des exemples Java, consultez Exemples de SDK Java Amazon SQS.

Pour connaître les bonnes pratiques d'utilisation des files d'attente FIFO, consultez Recommandations supplémentaires pour les files d'attente FIFO Amazon SQS et Recommandations pour les files d'attente Amazon SQS standard et FIFO.

Logique de distribution des files d'attente FIFO dans Amazon SQS

Les concepts suivants peuvent vous aider à mieux comprendre l'envoi et la réception de messages à partir de FIFO.

Envoi de messages

Si plusieurs messages sont envoyés à la suite vers une file d'attente FIFO, chacun avec un ID de déduplication du message différent, Amazon SQS stocke les messages et confirme la transmission. Puis, chaque message peut être reçu et traité dans l'ordre exact dans lequel les messages ont été transmis.

Dans les files d'attente FIFO, les messages sont classés selon l'ID de groupe de messages. Si plusieurs hôtes (ou threads différents sur le même hôte) envoient des messages avec le même ID de groupe de messages à une file d'attente FIFO, Amazon SQS stocke les messages dans l'ordre dans lequel ils arrivent pour le traitement. Pour qu'Amazon SQS conserve l'ordre d'envoi et de réception des messages, chaque producteur doit utiliser un ID de groupe de messages unique pour envoyer tous ses messages.

La logique de la file d'attente FIFO s'applique uniquement par ID de groupe de messages. Chaque ID de groupe de messages représente un groupe de messages classés différent au sein d'une file d'attente Amazon SQS. Pour chaque ID de groupe de messages, tous les messages sont envoyés et reçus en suivant rigoureusement l'ordre établi. Cependant, les messages aux ID de groupe de messages différents pourront être envoyés et reçus dans le désordre. Vous devez associer un ID de groupe de messages à un message. Si vous ne spécifiez pas d'ID de groupe de messages, l'action échoue. Si vous avez besoin d'un seul groupe de messages classés, indiquez le même ID de groupe de messages pour les messages envoyés à la file d'attente FIFO.

Réception de messages

Vous ne pouvez pas demander à recevoir des messages avec un ID de groupe de messages spécifique.

Lorsque vous recevez des messages d'une file d'attente FIFO avec plusieurs ID de groupe de messages, Amazon SQS tente d'abord de renvoyer autant de messages avec le même ID de groupe de messages que possible. Cela permet aux autres utilisateurs de traiter les messages avec un ID de groupe de messages différent. Lorsque vous recevez un message avec un ID de groupe de messages, aucun autre message correspondant au même ID de groupe de messages n'est renvoyé, sauf si vous supprimez le message ou s'il devient visible.

> **Note**
>
> Il est possible de recevoir jusqu'à 10 messages lors d'un seul appel en utilisant le paramètre de demande `MaxNumberOfMessages` de l'action `ReceiveMessage`. Ces messages conservent leur ordre FIFO et peuvent avoir le même ID de groupe de messages. Par conséquent, s'il y a moins de 10 messages disponibles avec le même ID de groupe de messages, vous pouvez recevoir des messages provenant d'un autre ID de groupe, dans le même lot de 10 messages, mais toujours dans l'ordre FIFO.

Multiples nouvelles tentatives

Les files d'attente FIFO permettent au producteur ou au consommateur d'effectuer plusieurs tentatives :

- Si le producteur détecte l'échec d'une action `SendMessage`, il peut réessayer d'en envoyer autant de fois que nécessaire, en utilisant le même identifiant de déduplication des messages. En supposant que le producteur reçoive au moins un accusé de réception avant l'expiration de l'intervalle de déduplication, les tentatives multiples n'affectent pas l'ordre des messages et n'introduisent pas de doublons.
- Si le consommateur détecte l'échec d'une action `ReceiveMessage`, il peut réessayer autant de fois que nécessaire, en utilisant le même identifiant de tentative de demande de réception. En supposant que le consommateur reçoive au moins un accusé de réception avant l'expiration du délai de visibilité, les tentatives multiples n'ont aucune incidence sur l'ordre des messages.
- Lorsque vous recevez un message avec un ID de groupe de messages, aucun autre message correspondant au même ID de groupe de messages n'est renvoyé, sauf si vous supprimez le message ou s'il devient visible.

Commande des messages de file d'attente FIFO dans Amazon SQS

La file d'attente FIFO améliore et complète la file d'attente standard. Les fonctions les plus importantes de ce type de file d'attente sont la remise FIFO (First-In First-Out) et le traitement en une seule fois :

- L'ordre dans lequel les messages sont envoyés et reçus est strictement préservé et un message est délivré une fois et reste indisponible jusqu'à ce qu'un consommateur le traite et le supprime.
- Aucun doublon n'est ajouté à la file d'attente.

En outre, les files d'attente FIFO prennent en charge les groupes de messages permettant à plusieurs groupes de messages classés de se trouver au sein d'une seule file d'attente. Il n'y a pas de quota quant au nombre de groupes de messages dans une file d'attente FIFO.

Traitement effectué en une seule fois dans Amazon SQS

Contrairement aux files d'attente standard, les files d'attente FIFO n'introduisent pas de messages en double. Les files d'attente FIFO vous aident à éviter d'envoyer des doublons à une file d'attente. Si vous réessayez l'action `SendMessage` dans un délai de déduplication de 5 minutes, Amazon SQS n'ajoute pas de doublons dans la file d'attente.

Pour configurer une déduplication, vous devez effectuer l'une des actions suivantes :

- Activer la déduplication basée sur le contenu. Amazon SQS reçoit l'ordre d'utiliser un hachage SHA-256 pour générer l'ID de déduplication du message en utilisant le corps de celui-ci, mais pas ses attributs. Pour plus d'informations, consultez la documentation sur les actions `CreateQueue`, `GetQueueAttributes` et `SetQueueAttributes` dans la Référence d'API Amazon Simple Queue Service.
- Fournir explicitement l'ID de déduplication du message (ou afficher le numéro de séquence) pour le message. Pour plus d'informations, consultez la documentation sur les actions `SendMessage`, `SendMessageBatch` et `ReceiveMessage` dans la Référence d'API Amazon Simple Queue Service.

Passage d'une file d'attente standard à une file d'attente FIFO dans Amazon SQS

Si l'une de vos applications utilise des files d'attente standard et que vous souhaitez tirer parti des fonctions du classement ou du traitement unique des files d'attente FIFO, vous devez configurer la file d'attente et l'application correctement.

> **Note**
>
> Vous ne pouvez pas convertir une file d'attente standard existante en file d'attente FIFO. Vous devez créer une nouvelle file d'attente FIFO pour votre application ou supprimer la file d'attente standard et la recréer en tant que file d'attente FIFO.

Utilisez la liste de contrôle suivante afin de vérifier que l'application fonctionne correctement avec une file d'attente FIFO :

- Utilisez le mode débit élevé recommandé pour la file d'attente FIFO afin d'augmenter le débit. Pour en savoir plus sur les quotas de messages, consultez Quotas de messages Amazon SQS.
- Les files d'attente FIFO ne prennent pas en charge les retards par message, uniquement les retards par file d'attente. Si l'application définit la même valeur du paramètre `DelaySeconds` sur chaque message, vous devez la modifier pour supprimer le retard par message et définir plutôt le paramètre `DelaySeconds` sur l'ensemble de la file d'attente.
- Le groupe de messages est une fonctionnalité FIFO unique qui permet aux clients de traiter les messages en parallèle tout en conservant leurs commandes respectives. Les clients organisent les messages en groupes de messages en spécifiant un ID de groupe de messages. Les groupes de messages sont souvent basés sur une dimension professionnelle pour une charge de travail donnée. Pour mieux mettre à l'échelle les files d'attente FIFO, utilisez une dimension professionnelle plus précise pour l'ID des messages. Plus le nombre d'identifiants de groupes de messages auxquels vous distribuez des messages est important, plus le nombre de messages mis à disposition par FIFO pour la consommation est important.
- Avant d'envoyer des messages à une file d'attente FIFO, confirmez ce qui suit :
  - Si l'application peut envoyer des messages avec des corps identiques, vous pouvez la modifier pour fournir un ID de déduplication du message unique pour chaque message envoyé.
  - Si l'application envoie des messages avec un corps unique, vous pouvez activer la déduplication basée sur le contenu.

- Vous n'avez pas besoin d'effectuer de modifications de code pour votre consommateur. Toutefois, si le traitement des messages prend beaucoup de temps et que le délai de visibilité est défini sur une valeur élevée, vous devriez ajouter un ID de tentative de demande de réception à chaque action `ReceiveMessage`. Cela vous permet de recommencer les tentatives de réception en cas de défaillance de la mise en réseau et empêche les files d'attente de s'interrompre en raison d'un échec des tentatives de réception.

Pour plus d'informations, veuillez consulter la Référence d'API Amazon Simple Queue Service.

Débit élevé pour les files d'attente FIFO dans Amazon SQS

Les files d'attente FIFO à haut débit d'Amazon SQS gèrent efficacement le débit élevé de messages tout en maintenant un ordre strict des messages, garantissant ainsi la fiabilité et l'évolutivité des applications traitant de nombreux messages. Cette solution est idéale pour les scénarios exigeant à la fois un débit élevé et une livraison ordonnée des messages.

Les files d'attente FIFO à haut débit Amazon SQS ne sont pas nécessaires dans les scénarios où un ordre strict des messages n'est pas crucial et où le volume de messages entrants est relativement faible ou sporadique. Par exemple, si vous avez une application à petite échelle qui traite des messages peu fréquents ou non séquentiels, la complexité et les coûts supplémentaires associés aux files d'attente FIFO à haut débit peuvent ne pas être justifiés. En outre, si votre application n'a pas besoin des capacités de débit améliorées fournies par les files d'attente FIFO à haut débit, opter pour une file d'attente Amazon SQS standard peut s'avérer plus rentable et plus simple à gérer.

Pour améliorer la capacité de demande dans les files d'attente FIFO à haut débit, il est recommandé d'augmenter le nombre de groupes de messages. Pour plus d'informations sur les quotas de messages à débit élevé, consultez Quotas du service Amazon SQS dans le Référence générale d'Amazon Web Services

Pour plus d'informations sur les quotas par file d'attente et les stratégies de distribution des données, reportez-vous aux sections Quotas de messages Amazon SQS etPartitions et distribution de données pour un débit élevé pour les files d'attente FIFO SQS.

Rubriques

- Cas d'utilisation du débit élevé pour les files d'attente FIFO Amazon SQS
- Partitions et distribution de données pour un débit élevé pour les files d'attente FIFO SQS
- Activez un débit élevé pour les files d'attente FIFO dans Amazon SQS

Cas d'utilisation du débit élevé pour les files d'attente FIFO Amazon SQS

Les cas d'utilisation suivants mettent en évidence les diverses applications des files d'attente FIFO à haut débit, démontrant leur efficacité dans tous les secteurs et dans tous les scénarios :

1. Traitement des données en temps réel : les applications traitant des flux de données en temps réel, tels que le traitement d'événements ou l'ingestion de données de télémétrie, peuvent bénéficier de files d'attente FIFO à haut débit pour gérer l'afflux continu de messages tout en préservant leur ordre pour une analyse précise.
2. Traitement des commandes de commerce électronique : sur les plateformes de commerce électronique où le maintien de l'ordre des transactions des clients est essentiel, les files d'attente FIFO à haut débit garantissent que les commandes sont traitées de manière séquentielle et sans délai, même pendant les périodes de pointe des achats.
3. Services financiers : les institutions financières traitant des transactions ou des transactions à haute fréquence s'appuient sur des files d'attente FIFO à haut débit pour traiter les données de marché et les transactions avec une latence minimale, tout en respectant des exigences réglementaires strictes en matière de commande des messages.
4. Streaming multimédia : les plateformes de streaming et les services de distribution multimédia utilisent des files d'attente FIFO à haut débit pour gérer la diffusion des fichiers multimédia et du contenu en streaming, garantissant ainsi des expériences de lecture fluides aux utilisateurs tout en maintenant le bon ordre de diffusion du contenu.

Partitions et distribution de données pour un débit élevé pour les files d'attente FIFO SQS

Amazon SQS stocke les données des files d'attente FIFO dans des partitions. Une partition est une allocation de stockage pour une file d'attente qui est automatiquement répliquée sur plusieurs zones de disponibilité au sein d'une AWS région. Vous ne gérez pas les partitions. Amazon SQS gère la gestion des partitions.

Pour les files d'attente FIFO, Amazon SQS modifie le nombre de partitions dans une file d'attente dans les situations suivantes :

- Si le taux de demandes actuel approche ou dépasse la limite de prise en charge des partitions existantes, des partitions supplémentaires sont allouées jusqu'à ce que la file d'attente atteigne le quota régional. Pour plus d'informations sur les quotas, consultez Quotas de messages Amazon SQS.

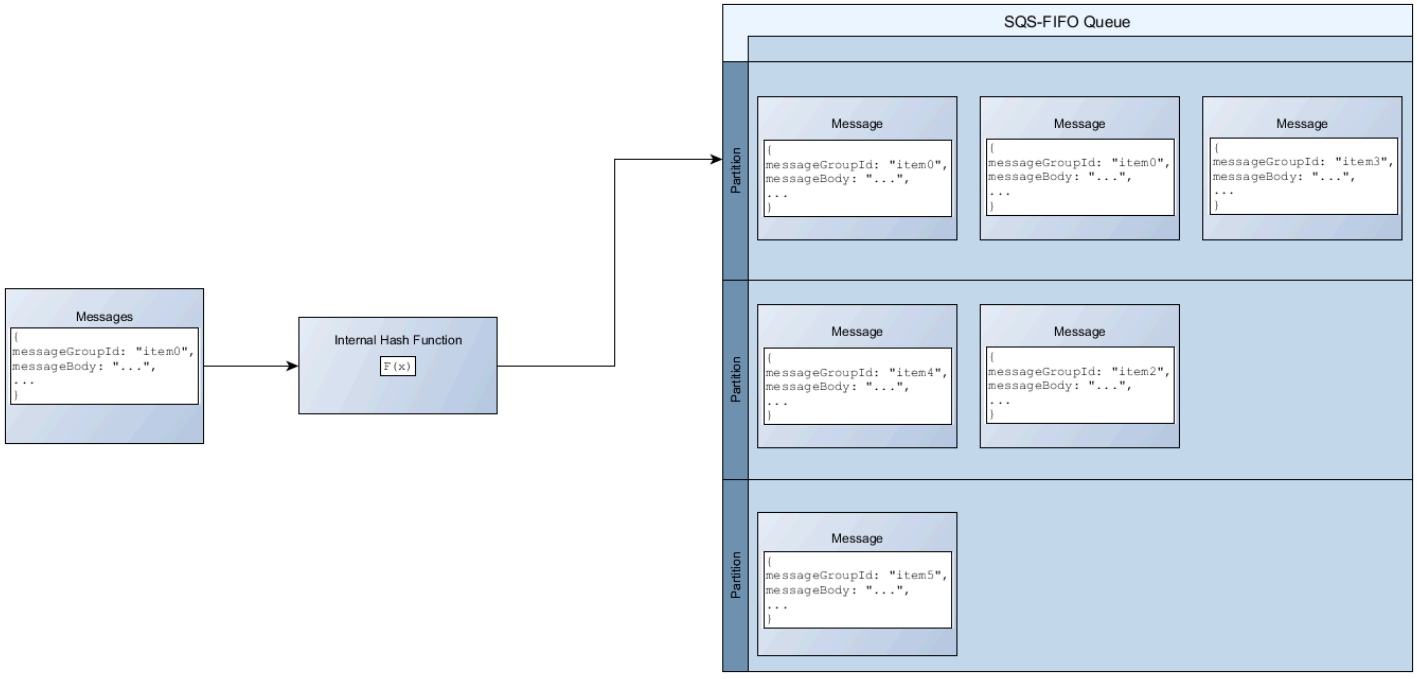
- Si les partitions actuelles sont peu utilisées, le nombre de partitions peut être réduit.

La gestion de la partition s'effectue automatiquement à l'arrière-plan et est transparente pour vos applications. Votre file d'attente et vos messages sont disponibles à tout moment.

Distribution des données par ID de groupe de messages

Pour ajouter un message à une file d'attente FIFO, Amazon SQS utilise la valeur de l'ID de groupe de messages de chaque message comme entrée dans une fonction de hachage interne. La valeur de sortie de la fonction de hachage détermine la partition dans laquelle le message sera stocké.

Le diagramme suivant illustre une file d'attente qui s'étend sur plusieurs partitions. L'ID du groupe de messages de la file d'attente est basé sur le numéro d'élément. Amazon SQS utilise sa fonction de hachage pour déterminer où stocker un nouvel élément, en l'occurrence, en fonction de la valeur de hachage de la chaîne `item0`. Notez que les éléments sont stockés dans le même ordre dans lequel ils sont ajoutés à la file d'attente. L'emplacement de chaque élément est déterminé par la valeur de hachage de son ID de groupe de messages.



> **Note**
>
> Amazon SQS est optimisé pour une distribution uniforme des éléments entre les partitions d'une file d'attente FIFO, quel que soit le nombre de partitions. AWS recommande d'utiliser

des identifiants de groupes de messages pouvant comporter un grand nombre de valeurs distinctes.

Optimisation de l'utilisation des partitions

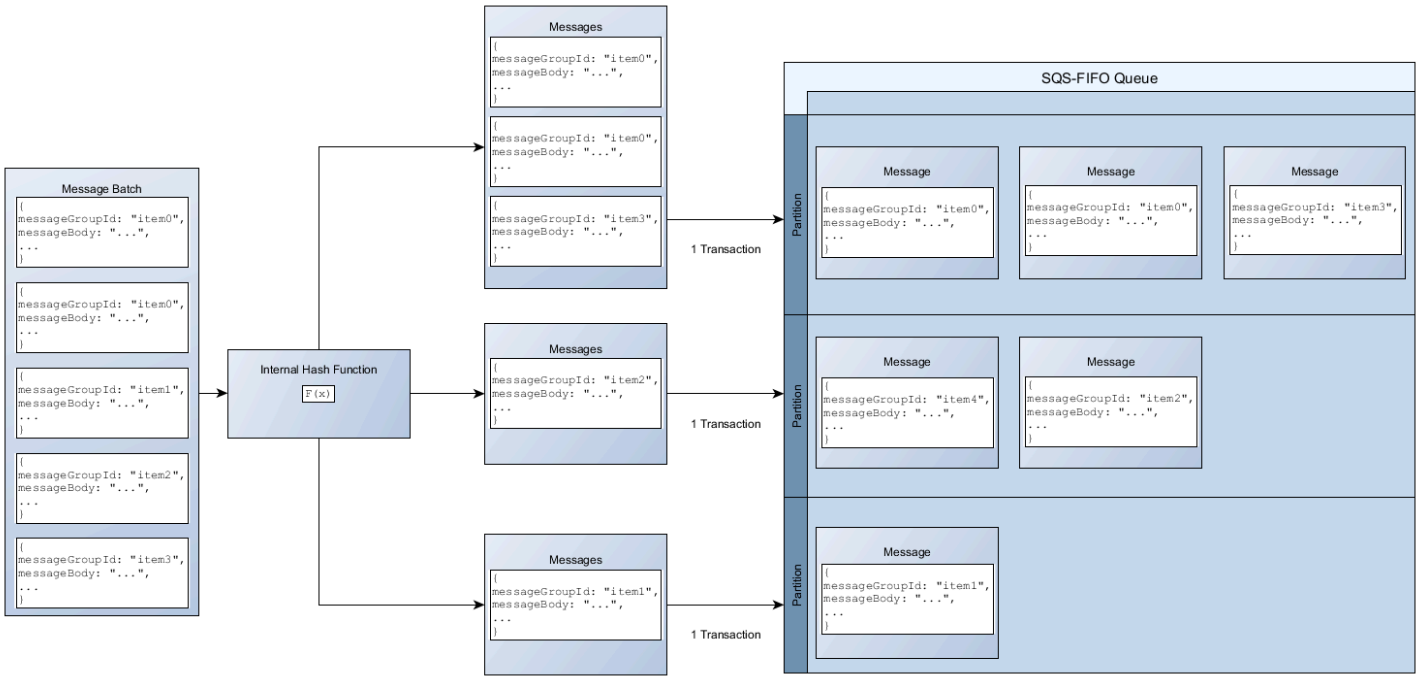
Chaque partition peut prendre en charge jusqu'à 3 000 messages par seconde avec le traitement par lots, ou jusqu'à 300 messages par seconde pour les opérations d'envoi, de réception et de suppression dans les régions prises en charge. Pour plus d'informations sur les quotas de messages à débit élevé, consultez Quotas du service Amazon SQS dans le Référence générale d'Amazon Web Services

Lorsque vous utilisez des API par lots, chaque message est acheminé selon le processus décrit dans Distribution des données par ID de groupe de messages. Les messages acheminés vers la même partition sont regroupés et traités en une seule transaction.

Pour optimiser l'utilisation des partitions pour l'`SendMessageBatch`API, il est AWS recommandé de regrouper les messages avec les mêmes identifiants de groupe de messages lorsque cela est possible.

Pour optimiser l'utilisation des partitions pour les `ChangeMessageVisibilityBatch` API `DeleteMessageBatch` et, il est AWS recommandé d'utiliser des `ReceiveMessage` requêtes dont le `MaxNumberOfMessages` paramètre est défini sur 10 et de regrouper par lots les descripteurs de réception renvoyés par une seule demande. `ReceiveMessage`

Dans l'exemple suivant, un lot de messages avec différents identifiants de groupes de messages est envoyé. Le lot est divisé en trois groupes, chacun étant pris en compte dans le quota de la partition.



> **Note**
>
> Amazon SQS garantit uniquement que les messages dotés de la fonction de hachage interne du même ID de groupe de messages sont regroupés au sein d'une demande groupée. En fonction du résultat de la fonction de hachage interne et du nombre de partitions, les messages portant des ID de groupe de messages différents peuvent être regroupés. Comme la fonction de hachage ou le nombre de partitions peut changer à tout moment, les messages groupés à un moment donné ne le seront peut-être pas ultérieurement.

Activez un débit élevé pour les files d'attente FIFO dans Amazon SQS

Vous pouvez activer le débit élevé pour toute file d'attente FIFO nouvelle ou existante. Cette fonctionnalité inclut trois nouvelles options lorsque vous créez et modifiez des files d'attente FIFO :

- Activer le FIFO à haut débit : met à disposition du débit plus élevé pour les messages de la file d'attente FIFO actuelle.
- Portée de la déduplication : spécifie si la déduplication a lieu au niveau de la file d'attente ou du groupe de messages.
- Limite de débit FIFO : spécifie si le quota de débit des messages de la file d'attente FIFO est défini au niveau de la file d'attente ou du groupe de messages.

Pour activer un débit élevé pour une file d'attente FIFO (console)

1. Commencez à créer ou à modifier une file d'attente FIFO.
2. Lorsque vous spécifiez les options pour la file d'attente, choisissez Activer le FIFO à haut débit.

   L'activation du débit élevé pour les files d'attente FIFO définit les options associées comme suit :

   - La Portée de la déduplication est définie sur Groupe de messages, le paramètre requis pour utiliser le débit élevé pour les files d'attente FIFO.
   - La Limite de débit FIFO est définie sur Par ID de groupe de messages, le paramètre requis pour utiliser le débit élevé pour les files d'attente FIFO.

   Si vous modifiez l'un des paramètres requis pour utiliser le débit élevé pour les files d'attente FIFO, le débit normal est effectif pour la file d'attente et la déduplication se produit comme indiqué.
3. Continuez à spécifier toutes les options pour la file d'attente. Lorsque vous avez terminé, choisissez Créer une file d'attente ou Enregistrer.

Après avoir créé ou modifié la file d'attente FIFO, vous pouvez lui envoyer des messages, en recevoir et en supprimer, le tout à un TPS plus élevé. Pour connaître les quotas de débit élevé, consultez la section Débit des messages dans Quotas de messages Amazon SQS.

Termes clés d'Amazon SQS

Les termes clés suivants peuvent vous aider à mieux comprendre les fonctionnalités des files d'attente FIFO. Pour plus d'informations, consultez la Référence d'API Amazon Simple Queue Service.

ID de déduplication du message

Jeton utilisé pour la déduplication des messages envoyés. Si un message avec un ID de déduplication de message particulier est correctement envoyé, tous les messages envoyés avec le même ID de déduplication de message sont correctement acceptés, mais ne sont pas remis pendant l'intervalle de déduplication de 5 minutes.

> **Note**
>
> Amazon SQS continue de suivre l'ID de déduplication du message même après la réception et la suppression du message.

ID de groupe de messages

La balise qui spécifie qu'un message appartient à un groupe de messages spécifique. Les messages appartenant au même groupe de messages sont toujours traités un à la fois, dans un ordre strict par rapport au groupe de messages (toutefois, les messages appartenant à des groupes de messages différents peuvent être traités dans le désordre).

ID de tentative de demande de réception

Jeton utilisé pour la déduplication des appels `ReceiveMessage`.

Numéro de séquence

Le grand numéro non consécutif qu'Amazon SQS attribue à chaque message.

Compatibilité FIFO dans Amazon SQS

Clients

Le client asynchrone en mémoire tampon Amazon SQS ne prend actuellement pas en charge les files d'attente FIFO.

Services

Si votre application utilise plusieurs AWS services, ou une combinaison de AWS services externes, il est important de savoir quelle fonctionnalité de service ne prend pas en charge les files d'attente FIFO.

Certains services AWS ou services externes qui envoient des notifications à Amazon SQS peuvent ne pas être compatibles avec les files d'attente FIFO, même si vous pouvez définir une file d'attente FIFO comme cible.

Les fonctionnalités des AWS services suivantes ne sont actuellement pas compatibles avec les files d'attente FIFO :

- Notifications d'événements Amazon S3
- Hooks de cycle de vie autoscaling
- AWS IoT Actions relatives aux règles
- AWS Lambda Files d'attente de lettres mortes

Pour plus d'informations sur la compatibilité d'autres services avec les files d'attente FIFO, consultez la documentation de votre service.

Identifiants de file d'attente et de message FIFO dans Amazon SQS

Cette section décrit les identifiants des files d'attente FIFO. Ces identifiants peuvent vous aider à trouver et à manipuler des files d'attente et des messages spécifiques.

Rubriques

- Identifiants pour les files d'attente FIFO dans Amazon SQS
- Identifiants supplémentaires pour les files d'attente FIFO Amazon SQS

Identifiants pour les files d'attente FIFO dans Amazon SQS

Pour plus d'informations sur les identifiants suivants, consultez la Référence d'API Amazon Simple Queue Service.

Nom et URL de la file d'attente

Lorsque vous créez une file d'attente, vous devez indiquer un nom unique pour le compte et la région AWS . Amazon SQS attribue à chaque file d'attente que vous créez un identifiant appelé URL de file d'attente qui inclut le nom de la file d'attente et les autres composants Amazon SQS. Chaque fois que vous souhaitez effectuer une action au niveau d'une file d'attente, vous devez fournir cette URL.

Le nom d'une file d'attente FIFO doit se terminer par le suffixe `.fifo`. Le suffixe est pris en compte dans le quota de 80 caractères pour les noms de file d'attente. Pour déterminer si une file d'attente est de type FIFO, vous pouvez vérifier si son nom se termine par le suffixe.

Voici l'URL d'une file d'attente FIFO nommée `MyQueue` appartenant à un utilisateur possédant le numéro `123456789012` de compte AWS.

```
https://sqs.us-east-2.amazonaws.com/123456789012/MyQueue.fifo
```

Vous pouvez extraire l'URL d'une file d'attente par programmation en listant vos files d'attente et en analysant la chaîne qui suit le numéro de compte. Pour plus d'informations, consultez `ListQueues`.

ID de message

Chaque message reçoit un ID de message attribué par le système qu'Amazon SQS vous renvoie dans la réponse `SendMessage`. Cet identifiant est utile pour identifier les messages. La longueur maximale d'un ID de message est de 100 caractères.

Descripteur de réception

Chaque fois que vous recevez un message d'une file d'attente, vous recevez un descripteur de réception correspondant. Cette gestion est associée à la réception du message, et non au message lui-même. Pour supprimer le message ou pour en modifier la visibilité, vous devez fournir le descripteur de réception (et non l'ID du message). C'est pourquoi vous devez toujours recevoir un message avant de pouvoir le supprimer (vous ne pouvez pas placer un message dans la file d'attente, puis le rappeler). La longueur maximale d'un descripteur de réception est de 1 024 caractères.

> **Important**
>
> Si vous recevez un message plusieurs fois, chaque fois que vous le recevez, vous obtenez un descripteur de réception différent. Lorsque vous demandez la suppression du message, vous devez fournir le descripteur de réception le plus récent. Dans le cas contraire, la suppression peut ne pas fonctionner.

Voici un exemple de descripteur de réception (réparti sur trois lignes).

```
MbZj6wDWli+JvwwJaBV+3dcjk2YW2vA3+STFFljTM8tJJg6HRG6PYSasuWXPJB+Cw
Lj1FjgXUv1uSj1gUPAWV66FU/WeR4mq2OKpEGYWbnLmpRCJVAyeMjeU5ZBdtcQ+QE
auMZc8ZRv37sIW2iJKq3M9MFx1YvV11A2x/KSbkJ0=
```

Identifiants supplémentaires pour les files d'attente FIFO Amazon SQS

Pour plus d'informations sur les identifiants suivants, consultez Traitement effectué en une seule fois dans Amazon SQS et la Référence d'API Amazon Simple Queue Service.

ID de déduplication du message

Jeton utilisé pour la déduplication des messages envoyés. Si un message avec un ID de déduplication de message particulier est correctement envoyé, tous les messages envoyés avec le même ID de déduplication de message sont correctement acceptés, mais ne sont pas remis pendant l'intervalle de déduplication de 5 minutes.

ID de groupe de messages

La balise qui spécifie qu'un message appartient à un groupe de messages spécifique. Les messages appartenant au même groupe de messages sont toujours traités un à la fois, dans un ordre strict par rapport au groupe de messages (toutefois, les messages appartenant à des groupes de messages différents peuvent être traités dans le désordre).

Numéro de séquence

Le grand numéro non consécutif qu'Amazon SQS attribue à chaque message.

Quotas Amazon SQS

Cette rubrique répertorie les quotas au sein d'Amazon Simple Queue Service (Amazon SQS).

Rubriques

Quotas de file d'attente FIFO Amazon SQS

Quotas Amazon SQS

Le tableau suivant répertorie les quotas relatifs aux files d'attente FIFO.

Quota	Description
File d'attente à retardement	Le délai (minimum) par défaut pour une file d'attente est de 0 seconde. La valeur maximale est de 15 minutes.
Files d'attente répertoriées	1000 files d'attente par demande `ListQueues` .
Durée d'attente pour interrogation longue	Le temps d'attente maximal pour la recherche prolongée est de 20 secondes.
Groupes de messages	Il n'y a pas de quota quant au nombre de groupes de messages dans une file d'attente FIFO.
Messages par file d'attente (en attente)	Le nombre de messages qu'une file d'attente Amazon SQS peut stocker est illimité.
Messages par file d'attente (en transit)	Pour les files d'attente FIFO, il peut y avoir un maximum de 20 000 messages en cours (reçus depuis une file d'attente par un consommateur, mais pas encore supprimés de la file d'attente). Si vous atteignez ce quota, Amazon SQS ne renvoie aucun message d'erreur.

Quota	Description
Nom de la file d'attente	Le nom d'une file d'attente FIFO doit se terminer par le suffixe `.fifo`. Le suffixe est pris en compte dans le quota de 80 caractères pour les noms de file d'attente . Pour déterminer si une file d'attente est de type FIFO, vous pouvez vérifier si son nom se termine par le suffixe.
Balise de file d'attente	Nous vous déconseillons d'ajouter plus de 50 balises à une file d'attente. Le balisage prend en charge les caractères Unicode en UTF-8.
	La balise `Key` est obligatoire, mais la balise `Value` est facultative.
	La balise `Key` et la balise `Value` sont sensibles à la casse.
	La balise `Key` et la balise `Value` peuvent inclure des caractères alphanumériques Unicode en UTF-8 et des espaces blancs. Les caractères spéciaux suivants sont acceptés : `_ . : / = + - @`
	La balise `Key` ou `Value` ne doit pas inclure le préfixe réservé `aws:` (vous ne pouvez pas supprimer les clés de balise ou les valeurs ayant ce préfixe).
	La longueur `Key` de balise maximale est de 128 caractère s Unicode en UTF-8. La balise `Key` ne doit pas être vide ou null.
	La longueur `Value` de balise maximale est de 256 caractères Unicode en UTF-8. La balise `Value` peut être vide ou null.
	Les actions de balisage sont limitées à 30 TPS par action. Compte AWS Si votre application nécessite un débit plus élevé, soumettez une demande.

Quotas de messages Amazon SQS

Le tableau suivant répertorie les quotas relatifs aux messages.

Quota	Description
ID de message par lots	Un identifiant de message groupé peut comporter jusqu'à 80 caractères. Les caractères suivants sont acceptés : caractères alphanumériques, tirets (-) et traits de soulignement (_).
Attributs de message	Un message peut contenir jusqu'à 10 attributs de métadonnées.
Traitement par lots des messages	Une demande de traitement par lots de messages peut inclure un maximum de 10 messages. Pour plus d'informations, consultez Configuration du client AmazonSQSBufferedAsync dans la section Actions groupées Amazon SQS.
Contenu des messages	Un message peut contenir uniquement du texte XML ou JSON et du texte non formaté. Les caractères Unicode suivants sont acceptés : `#x9` | `#xA` | `#xD` | `#x20` to `#xD7FF` | `#xE000` à `#xFFFD` | `#x10000` à `#x10FFFF` Tous les caractères non inclus dans cette liste sont refusés. Pour plus d'informations, consultez la spécification W3C en matière de caractères.
ID de groupe de messages	Utilisez les messages en attente pour éviter d'accumuler un journal volumineux des messages en attente avec le même ID de groupe de messages. `MessageGroupId` est obligatoire pour les files d'attente FIFO. Vous ne pouvez pas l'utiliser pour les files d'attente standard.

Quota	Description
	Vous devez associer un message `MessageGroupId` non vide à un message. Si vous ne spécifiez pas de `MessageGroupId` , l'action échoue. La longueur maximale de `MessageGroupId` est de 128 caractères. Valeurs valides : caractères alphanumé riques et ponctuation `(!"#$%&'()*+,-./:;<=>? @[\]^_`{|}~)` .
Conservation des messages	Par défaut, un message est conservé pendant 4 jours. La durée minimale est de 60 secondes (1 minute). La durée maximale est de 1 209 600 secondes (14 jours).
Débit de message	Les files d'attente standard prennent en charge un nombre presque illimité d'appels d'API par seconde, par action d'API (`SendMessage` , `ReceiveMessage` ou `DeleteMessage`).
	Files d'attente FIFO • Les files d'attente FIFO prennent en charge un quota de 300 transactions par seconde, par action d'API (`SendMessage` , `ReceiveMessage` et `DeleteMes sage`). • Si vous utilisez le traitement par lots, les files d'attente FIFO prennent en charge jusqu'à 3 000 messages par seconde, par action d'API (`SendMessage` , `ReceiveMessage` ou `DeleteMessage`). Les 3 000 messages par seconde représentent 300 appels d'API, chacun avec un lot de 10 messages.

Quota	Description
	Débit élevé pour les files d'attente FIFO • Sans le traitement par lots (`SendMessage` , `ReceiveMessage` et `DeleteMessage`), les files d'attente FIFO à débit élevé peuvent traiter jusqu'à 70 000 transactions par seconde, par action d'API dans les régions USA Est (Virginie du Nord), USA Ouest (Oregon) et Europe (Irlande). • Pour les régions USA Est (Ohio) et Europe (Francfor t), le débit par défaut est de 18 000 transactions par seconde et par action d'API. • Pour les régions de l'Asie-Pacifique (Mumbai), de l'Asie-Pacifique (Singapour), de l'Asie-Pacifique (Sydney) et de l'Asie-Pacifique (Tokyo), le débit par défaut est de 9 000 transactions par seconde et par action d'API. • Pour l'Europe (Londres) et l'Amérique du Sud (São Paulo), le débit par défaut est de 4 500 transactions par seconde et par action d'API. • Pour un débit maximal, augmentez le nombre d'ID de groupes de messages que vous utilisez pour les messages envoyés sans traitement par lots. • Vous pouvez faire passer le débit à 700 000 messages par seconde en utilisant des API de traitement par lots (`SendMessageBatch` et `DeleteMes sageBatch`) dans les régions USA Est (Virginie du Nord), USA Ouest (Oregon) et Europe (Irlande) . Les 700 000 messages par seconde représentent 70 000 transactions par seconde, chacune constituant un lot de 10 messages. Pour les régions Europe (Francfort) et USA Est (Ohio), vous pouvez atteindre un débit de 180 000 messages par seconde en utilisant des API de traitement par

Quota	Description
	lots. Les 180 000 messages par seconde représentent 18 000 transactions par seconde, chacune avec un lot de 10 messages. Pour les régions de l'Asie-Pacifique (Mumbai), de l'Asie-Pacifique (Singapour), de l'Asie-Pacifique (Sydney) et de l'Asie-Pacifique (Tokyo), vous pouvez obtenir jusqu'à 90 000 messages par seconde grâce au traitement par lots. Pour atteindre le débit maximal lors de l'utilisation de `SendMessageBatch` et `DeleteMessageBatch`, tous les messages d'une demande par lots doivent utiliser le même ID de groupe de messages. • Pour les régions d'Europe (Londres) et d'Amérique du Sud (São Paulo), vous pouvez obtenir jusqu'à 45 000 messages par seconde grâce au traitement par lots. Pour atteindre le débit maximal lors de l'utilisation de `SendMessageBatch` et `DeleteMessageBatch`, tous les messages d'une demande par lots doivent utiliser le même ID de groupe de messages. • Dans toutes les autres AWS régions, le débit maximal est de 2 400 (sans traitement par lots) ou 24 000 (avec traitement par lots) messages par seconde, par action d'API. • Pour demander une augmentation du quota au-delà de la limite de région, soumettez une demande d'assistance. • Pour plus d'informations, consultez Partitions et distribution de données pour un débit élevé pour les files d'attente FIFO SQS.
Temporisateur de message	Le délai (minimum) par défaut pour un message est de 0 seconde. La valeur maximale est de 15 minutes.

Quota	Description
Message size (Taille de message)	La taille minimale de message est de 1 octet (1 caractère). La taille maximale est de 262 144 octets (256 Kio). Pour envoyer des messages supérieurs à 256 KiB, vous pouvez utiliser la bibliothèque client étendue Amazon SQS pour Java et la bibliothèque client étendue Amazon SQS pour Python. Cette bibliothèque vous permet d'envoyer un message Amazon SQS qui contient une référence à une charge utile de message dans Amazon S3. La taille de la charge utile maximale est de 2 Go. **Note** Cette bibliothèque étendue ne fonctionne que pour les clients synchrones.
Délai de visibilité des messages	Le délai de visibilité par défaut d'un message est de 30 secondes. La valeur minimale est 0 seconde. La valeur maximale est 12 heures.
Informations sur les politiques	Le quota maximal est de 8 192 octets, 20 instructions, 50 mandataires ou 10 conditions. Pour plus d'informations, voir Quotas liés à la politique Amazon SQS.

Quotas liés à la politique Amazon SQS

Le tableau suivant répertorie les quotas relatifs aux stratégies.

Nom	Maximum
Octets	8 192
Conditions	10

Nom	Maximum
Mandataires	50
Instructions	20
Actions par instruction	7

Fonctionnalités et capacités d'Amazon SQS

Amazon SQS fournit les fonctionnalités et capacités suivantes :

Rubriques

Utilisation de files d'attente contenant des lettres mortes dans Amazon SQS

Amazon SQS prend en charge les files d'attente de lettres mortes (DLQ), que les files d'attente sources peuvent cibler pour les messages qui ne sont pas traités correctement. Les DLQ sont utiles pour le débogage de votre application car vous pouvez isoler les messages non consommés afin de déterminer pourquoi le traitement a échoué. Pour des performances optimales, il est recommandé de conserver la même file d'attente source et le DLQ dans la même Compte AWS région. Une fois que les messages se trouvent dans une file d'attente de lettres mortes, vous pouvez :

- Rechercher dans les journaux les exceptions qui ont pu entraîner le déplacement de messages dans une file d'attente de lettres mortes.
- Analysez le contenu des messages placés dans la file d'attente des lettres mortes pour diagnostiquer les problèmes liés à l'application.

- Déterminer si vous avez donné suffisamment de temps à votre consommateur pour traiter les messages.
- Déplacez les messages hors de la file d'attente des lettres mortes en utilisant le redrive de la file d'attente des lettres mortes.

Vous devez d'abord créer une nouvelle file d'attente avant de la configurer en tant que file d'attente contenant des lettres mortes. Pour plus d'informations sur la configuration d'une file d'attente de lettres mortes à l'aide de la console Amazon SQS, consultez Découvrez comment configurer une file d'attente contenant des lettres mortes à l'aide de la console Amazon SQS. Pour obtenir de l'aide sur les files d'attente de lettres mortes, par exemple sur la façon de configurer une alarme pour tout message déplacé vers une file d'attente de lettres mortes, consultez. Créez des alarmes pour les files d'attente de lettres mortes à l'aide d'Amazon CloudWatch

Utilisation de politiques pour les files d'attente de lettres mortes

Utilisez une politique de redrive pour spécifier le`maxReceiveCount`. `maxReceiveCount`C'est le nombre de fois qu'un consommateur peut recevoir un message depuis une file d'attente source avant qu'il ne soit déplacé vers une file d'attente de lettres mortes. Par exemple, si le paramètre `maxReceiveCount` est défini sur une valeur faible telle que 1, l'échec de réception d'un message entraîne son transfert dans la file d'attente des lettres mortes. Pour garantir la résilience de votre système face aux erreurs, définissez une valeur suffisamment élevée pour `maxReceiveCount` afin de permettre un nombre suffisant de tentatives.

La stratégie d'autorisation de redirection spécifie quelles files d'attente source peuvent accéder à la file d'attente de lettres mortes. Vous pouvez choisir d'autoriser toutes les files d'attente sources, d'autoriser des files d'attente sources spécifiques ou de refuser à toutes les files d'attente sources l'utilisation de la file d'attente de lettres mortes. Par défaut, toutes les files d'attente sources peuvent utiliser la file d'attente contenant des lettres mortes. Si vous choisissez d'autoriser des files d'attente spécifiques à l'aide de `byQueue` cette option, vous pouvez spécifier jusqu'à 10 files d'attente sources à l'aide de la file d'attente source Amazon Resource Name (ARN). Si vous spécifiez `denyAll`, la file d'attente ne peut pas être utilisée comme file d'attente de lettres mortes.

Comprendre les périodes de rétention des messages pour les files d'attente de lettres mortes

Pour les files d'attente standard, l'expiration d'un message est toujours basée sur son horodatage de mise en file d'attente d'origine. Lorsqu'un message est déplacé vers une

file d'attente de lettres mortes, l'horodatage de la mise en file d'attente reste inchangé. La `ApproximateAgeOfOldestMessage` métrique indique à quel moment le message a été placé dans la file d'attente des lettres mortes, et non à quel moment le message a été initialement envoyé. Supposons, par exemple, qu'un message passe 1 journée dans la file d'attente d'origine avant d'être déplacé vers une file d'attente de lettres mortes. Si la période de conservation de la file d'attente de lettres mortes est de 4 jours, le message est supprimé de la file d'attente de lettres mortes au bout de 3 jours et le paramètre `ApproximateAgeOfOldestMessage` est défini sur 3 jours. Il est donc recommandé de toujours définir la période de rétention d'une file d'attente de lettres mortes de manière à ce qu'elle soit plus longue que la période de rétention de la file d'attente d'origine.

Pour les files d'attente FIFO, l'horodatage de la mise en file d'attente est réinitialisé lorsque le message est déplacé vers une file d'attente de lettres mortes. La métrique `ApproximateAgeOfOldestMessage` indique à quel moment le message a été placé dans la file d'attente de lettres mortes. Dans le même exemple ci-dessus, le message est supprimé de la file d'attente de lettres mortes au bout de 4 jours et le paramètre `ApproximateAgeOfOldestMessage` est défini sur 4 jours.

Découvrez comment configurer une file d'attente contenant des lettres mortes à l'aide de la console Amazon SQS

Une file d'attente contenant des lettres mortes est une file que les files d'attente sources peuvent cibler pour les messages qui ne sont pas traités correctement. Pour plus d'informations, consultez Utilisation de files d'attente contenant des lettres mortes dans Amazon SQS .

Amazon SQS ne crée pas automatiquement la file d'attente de lettres mortes. Vous devez d'abord créer la file d'attente avant de l'utiliser en tant que file d'attente de lettres mortes. Pour obtenir des instructions sur la création d'une file d'attente à utiliser comme file d'attente lettre morte, voirCréation d'une file d'attente à l'aide de la console Amazon SQS.

La file d'attente de lettres mortes d'une file d'attente FIFO doit également être une file d'attente FIFO. De même, la file d'attente de lettres mortes d'une file d'attente standard doit également être une file d'attente standard.

Lorsque vous créez ou modifiez une file d'attente, vous pouvez configurer une file d'attente de lettres mortes.

Pour configurer une file d'attente de lettres mortes pour une file d'attente existante (console)

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.

2. Dans le volet de navigation, choisissez Files d'attente.
3. Sélectionnez une file d'attente et choisissez Modifier.
4. Accédez à la section File d'attente de lettres mortes et choisissez Activé.
5. Choisissez l'Amazon Resource Name (ARN) d'une file d'attente de lettres mortes existante que vous souhaitez associer à cette file d'attente source.
6. Pour configurer le nombre de fois qu'un message peut être reçu avant d'être envoyés à une file d'attente de lettres mortes, définissez Réceptions maximales sur une valeur comprise entre 1 et 1 000.
7. Lorsque vous avez fini de configurer la file d'attente de lettres mortes, choisissez Enregistrer.

   Une fois la file d'attente enregistrée, la console affiche la page Détails de votre file d'attente. Sur la page Détails, l'onglet File d'attente de lettres mortes affiche les Réceptions maximales et l'ARN de File d'attente de lettres mortes dans la File d'attente de lettres mortes.

Découvrez comment configurer le redrive d'une file d'attente en lettres mortes dans Amazon SQS

Vous pouvez utiliser le redrive de la file d'attente de lettres mortes pour déplacer les messages non consommés hors d'une file d'attente de lettres mortes existante. Par défaut, la redirection de la file d'attente de lettres mortes déplace les messages d'une file d'attente de lettres mortes vers une file d'attente source. Cependant, vous pouvez aussi configurer n'importe quelle autre file d'attente comme destination de redirection si les deux files d'attente sont du même type. Par exemple, si la file d'attente de lettres mortes est une file d'attente FIFO, la file d'attente de destination de redirection doit également être une file d'attente FIFO. En outre, vous pouvez configurer la vitesse de redirection pour définir la vitesse à laquelle Amazon SQS déplace les messages.

> **Note**
>
> Lorsqu'un message est déplacé d'une file d'attente FIFO vers une DLQ FIFO, l'ID de déduplication du message d'origine est remplacé par l'ID du message d'origine. Cela permet de s'assurer que la déduplication de la file d'attente de lettres mortes n'empêchera pas le stockage de deux messages indépendants partageant un identifiant de déduplication.

Les files d'attente de lettres mortes redirigent les messages dans l'ordre dans lequel ils sont reçus, en commençant par le message le plus ancien. Toutefois, la file d'attente de destination ingère les

messages redirigés, de même que les nouveaux messages provenant d'autres producteurs, en fonction de l'ordre dans lequel elle les reçoit. Par exemple, si un producteur envoie des messages à une file d'attente FIFO source alors qu'il reçoit simultanément des messages redirigés provenant d'une file d'attente de lettres mortes, les messages redirigés seront entrelacés avec les nouveaux messages du producteur.

> **Note**
>
> La tâche de redirection réinitialise la période de rétention. Tous les messages redirigés sont considérés comme de nouveaux messages avec un nouveau message `messageID` et `enqueueTime` sont affectés à des messages redirigés.

Rubriques

- Configuration d'un redrive de file d'attente en lettres mortes pour une file d'attente standard existante à l'aide de l'API Amazon SQS
- Configuration d'un redrive de file d'attente en lettres mortes pour une file d'attente standard existante à l'aide de la console Amazon SQS
- Configuration des autorisations de file d'attente pour la redirection de files d'attente de lettres mortes

Configuration d'un redrive de file d'attente en lettres mortes pour une file d'attente standard existante à l'aide de l'API Amazon SQS

Vous pouvez configurer un redrive de file d'attente contenant des lettres mortes à l'aide des actions`SendMessageBatch`,`ReceiveMessage`, et `DeleteMessageBatch` de l'API :

Action d'API	Description
`StartMessageMoveTask`	Démarre une tâche asynchrone pour déplacer les messages d'une file d'attente source spécifiée vers une file d'attente de destination spécifiée.

Action d'API	Description
ListMessageMoveTasks	Obtient les tâches de déplacement de messages les plus récentes (jusqu'à 10) dans une file d'attente source spécifique.
CancelMessageMoveTask	Annule une tâche de déplacement de message spécifiée. Un mouvement de message ne peut être annulé que lorsque le statut actuel est en cours d'exécution.

Configuration d'un redrive de file d'attente en lettres mortes pour une file d'attente standard existante à l'aide de la console Amazon SQS

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
2. Dans le volet de navigation, choisissez Files d'attente.
3. Choisissez le nom de la file d'attente que vous avez configurée en tant que file d'attente de lettres mortes.
4. Choisissez Démarrer la redirection de file d'attente de lettres mortes.
5. Dans Configuration de la redirection, pour Destination du message, effectuez l'une des opérations suivantes :
   - Pour rediriger les messages vers leur file d'attente source, choisissez Rediriger vers la ou les files d'attente source.
   - Pour rediriger les messages vers une autre file d'attente, choisissez Rediriger vers une destination personnalisée. Saisissez ensuite l'Amazon Resource Name (ARN) d'une file d'attente de destination existante.
6. Sous Paramètres de contrôle de la vitesse, choisissez l'une des options suivantes :
   - Système optimisé : redirigez les messages de la file d'attente de lettres mortes avec le nombre maximum de messages par seconde.
   - Vitesse maximale personnalisée : rediffusez les messages de la file d'attente de lettres mortes avec un débit maximal personnalisé de messages par seconde. Le débit maximal autorisé est de 500 messages par seconde.

- Il est recommandé de commencer par une petite valeur pour la vitesse maximale personnalisée et de vérifier que la file d'attente source n'est pas submergée de messages. Ensuite, augmentez progressivement la valeur de la vitesse maximale personnalisée, en continuant à surveiller l'état de la file d'attente source.

7. Lorsque vous avez fini de configurer la redirection de la file d'attente de lettres mortes, choisissez Rediriger les messages.

> **Important**
>
> Amazon SQS ne prend pas en charge le filtrage et la modification des messages lors de leur redirection à partir de la file d'attente de lettres mortes.
> Une tâche de redirection de file d'attente de lettres mortes peut s'exécuter pendant 36 heures au maximum. Amazon SQS prend en charge un maximum de 100 tâches de redirection actives par compte.

8. Si vous souhaitez annuler la tâche de redirection des messages, sur la page Détails de votre file d'attente, choisissez Annuler la redirection de la file d'attente de lettres mortes. Lorsque vous annulez la redirection d'un message en cours, tous les messages qui ont déjà été déplacés vers leur file d'attente de destination y resteront.

Configuration des autorisations de file d'attente pour la redirection de files d'attente de lettres mortes

Vous pouvez autoriser les utilisateurs à accéder à des actions spécifiques dans les files d'attente de lettres mortes en ajoutant des autorisations à votre stratégie. Les autorisations minimales requises pour la redirection d'une file d'attente de lettres mortes sont les suivantes :

Autorisations minimales	Méthodes d'API requises
Pour démarrer la redirection d'un message	• Ajoutez le `sqs:StartMessageMoveTask` , le `sqs:ReceiveMessage` , le `sqs:DeleteMessage` et le `sqs:GetQueueAttributes` de la file d'attente de lettres mortes. Si la file d'attente de lettres mortes ou la file d'attente source d'origine est chiffrée (également nommée file d'attente SSE), `kms:Decrypt` est également requis pour toute clé KMS utilisée pour chiffrer les messages.

Autorisations minimales	Méthodes d'API requises
	• Ajoutez le `sqs:SendMessage` de la file d'attente de destination. Si la file d'attente de destination est chiffrée, `kms:GenerateDataKey` et `kms:Decrypt` sont également requis.
Pour annuler un message en cours de redirection	• Ajoutez le `sqs:CancelMessageMoveTask`, le `sqs:ReceiveMessage`, le `sqs:DeleteMessage` et le `sqs:GetQueueAttributes` de la file d'attente de lettres mortes. Si la file d'attente de lettres mortes est chiffrée (également nommée file d'attente SSE), `kms:Decrypt` est également requis.
Pour afficher le statut du déplacement d'un message	• Ajoutez le `sqs:ListMessageMoveTasks` et le `sqs:GetQueueAttributes` de la file d'attente de lettres mortes.

Pour configurer les autorisations pour une paire de files d'attente chiffrées (une file d'attente source avec une file d'attente de lettres mortes)

Procédez comme suit pour configurer les autorisations minimales pour la redirection d'une file d'attente de lettres mortes :

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. Créez une stratégie avec les autorisations suivantes et attachez-la à votre utilisateur ou rôle IAM de connexion :

   - `sqs:StartMessageMoveTask`
   - `sqs:CancelMessageMoveTask`
   - `sqs:ListMessageMoveTasks`
   - `sqs:ListDeadLetterSourceQueues`
   - `sqs:ReceiveMessage`

- `sqs:DeleteMessage`
- `sqs:GetQueueAttributes`
- L'ARN `Resource` de la file d'attente de lettres mortes (par exemple, « arn:aws:sqs:*<DLQ_region>*:*<DLQ_accountId>*:*<DLQ_name>* »).
- `sqs:SendMessage`
- L'`Resource`ARN de la file d'attente de destination (par exemple, « arn:aws:sqs : *< DestQueue _region> : < _accountID> : < _name> «) DestQueue*. DestQueue
- `kms:Decrypt` : autorise l'action de déchiffrement.
- `kms:GenerateDataKey`
- Le ou les `Resource` ARN de toute clé de chiffrement KMS qui a été utilisée pour chiffrer les messages dans la file d'attente source d'origine (par exemple, « arn:aws:kms:*<region>*:*<accountId>*:key/*<keyId_used to encrypt the message body>* »).
- L'ARN de ressource de la clé de chiffrement KMS utilisée pour la file d'attente de destination de redirection (par exemple, « arn:aws:kms:*<region>*:*<accountId>*:key/*<keyId_used for the destination queue>* »).

Votre stratégie d'accès doit ressembler à ce qui suit :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sqs:StartMessageMoveTask",
        "sqs:CancelMessageMoveTask",
        "sqs:ListMessageMoveTasks",
        "sqs:ReceiveMessage",
        "sqs:DeleteMessage",
        "sqs:GetQueueAttributes",
        "sqs:ListDeadLetterSourceQueues"
      ],
      "Resource": "arn:aws:sqs:<DLQ_region>:<DLQ_accountId>:<DLQ_name>"
    },
    {
      "Effect": "Allow",
```

```
      "Action": "sqs:SendMessage",
      "Resource":
 "arn:aws:sqs:<DestQueue_region>:<DestQueue_accountId>:<DestQueue_name>"
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
      ],
      "Resource": "arn:aws:kms:<region>:<accountId>:key/<keyId>"
    }
  ]
}
```

Pour configurer les autorisations pour une paire de files d'attente non chiffrées (une file d'attente source avec une file d'attente de lettres mortes)

Procédez comme suit pour configurer les autorisations minimales pour une file d'attente de lettres mortes standard non chiffrée. Les autorisations minimales requises sont de recevoir, de supprimer et d'obtenir des attributs de la file d'attente de lettres mortes, et d'envoyer des attributs à la file d'attente source.

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. Créez une stratégie avec les autorisations suivantes et attachez-la à votre utilisateur ou rôle IAM de connexion :

   - `sqs:StartMessageMoveTask`
   - `sqs:CancelMessageMoveTask`
   - `sqs:ListMessageMoveTasks`
   - `sqs:ListDeadLetterSourceQueues`
   - `sqs:ReceiveMessage`
   - `sqs:DeleteMessage`
   - `sqs:GetQueueAttributes`

- L'ARN `Resource` de la file d'attente de lettres mortes (par exemple, « arn:aws:sqs:*<DLQ_region>*:*<DLQ_accountId>*:*<DLQ_name>* »).
- `sqs:SendMessage`
- *L'ResourceARN de la file d'attente de destination (par exemple, « arn:aws:sqs : < DestQueue _region> : < _accountID> : < _name> «) DestQueue. DestQueue*

Votre stratégie d'accès doit ressembler à ce qui suit :

```
{
"Version": "2012-10-17",
  "Statement": [
    {
"Effect": "Allow",
      "Action": [
        "sqs:StartMessageMoveTask",
        "sqs:CancelMessageMoveTask",
        "sqs:ListMessageMoveTasks",
        "sqs:ReceiveMessage",
        "sqs:DeleteMessage",
        "sqs:GetQueueAttributes",
        "sqs:ListDeadLetterSourceQueues"
      ],
      "Resource": "arn:aws:sqs:<DLQ_region>:<DLQ_accountId>:<DLQ_name>"
    },
    {
"Effect": "Allow",
      "Action": "sqs:SendMessage",
      "Resource":
 "arn:aws:sqs:<DestQueue_region>:<DestQueue_accountId>:<DestQueue_name>"
    }
  ]
}
```

CloudTrail exigences de mise à jour et d'autorisation pour le redrive de files d'attente de lettres mortes Amazon SQS

Le 8 juin 2023, Amazon SQS a introduit le redrive DLQ (Dead-Letter Queue) pour les SDK AWS et (CLI). AWS Command Line Interface Cette fonctionnalité vient s'ajouter au redrive DLQ déjà pris en charge pour la AWS console. Si vous avez déjà utilisé la AWS console pour rediffuser des messages de file d'attente en lettres mortes, vous pouvez être concerné par les modifications suivantes :

- CloudTrail changement de nom d'un événement pour le redrive de la file d'attente en lettres mortes
- Autorisations mises à jour pour la redirection de file d'attente de lettres mortes

CloudTrail changement de nom d'un événement

Le 15 octobre 2023, les noms des CloudTrail événements pour le redrive de la file d'attente de lettres mortes seront modifiés sur la console Amazon SQS. Si vous avez défini des alarmes pour ces CloudTrail événements, vous devez les mettre à jour dès maintenant. Les nouveaux noms d' CloudTrail événements pour DLQ redrive sont les suivants :

Nom de l'événement précédent	Nouveau nom de l'événement
`CreateMoveTask`	`StartMessageMoveTask`
`CancelMoveTask`	`CancelMessageMoveTask`

Autorisations mises à jour

Inclus dans la version du kit SDK et de la CLI, Amazon SQS a également mis à jour les autorisations de file d'attente pour la redirection de file d'attente de lettres mortes afin de respecter les bonnes pratiques de sécurité. Utilisez les types d'autorisation de file d'attente suivants pour rediriger les messages depuis vos files d'attente de lettres mortes.

1. Autorisations basées sur les actions (mise à jour pour les actions de l'API de file d'attente de lettres mortes)
2. Autorisations de stratégie Amazon SQS gérées
3. Stratégie d'autorisation utilisant un caractère générique sqs:*

> **Important**
>
> Pour utiliser la redirection de file d'attente de lettres mortes pour le kit SDK ou la CLI, vous devez disposer d'une stratégie d'autorisation de redirection de file d'attente de lettres mortes correspondant à l'une des options ci-dessus.

Si vos autorisations de file d'attente pour la redirection de file d'attente de lettres mortes ne correspondent pas à l'une des options ci-dessus, vous devez mettre à jour vos autorisations d'ici le 31 août 2023. D'ici le 31 août 2023, votre compte pourra rediriger les messages en utilisant les autorisations que vous avez configurées à l'aide de la console AWS uniquement dans les régions où vous avez déjà utilisé la redirection de file d'attente de lettres mortes. Par exemple, supposons que vous ayez un « compte A » à la fois dans les régions us-east-1 et eu-west-1. Le « compte A » était utilisé pour rediriger les messages sur la AWS console dans us-east-1 avant le 8 juin 2023, mais pas dans eu-west-1. Entre le 8 juin 2023 et le 31 août 2023, si les autorisations de politique du « compte A » ne correspondent pas à l'une des options ci-dessus, elles ne peuvent être utilisées que pour rediriger des messages sur la AWS console dans us-east-1, et non dans eu-west-1.

> **Important**
>
> Si vos autorisations de redirection de file d'attente de lettres mortes ne correspondent pas à l'une de ces options après le 31 août 2023, votre compte ne sera plus en mesure de rediriger les messages de file d'attente de lettres mortes à l'aide de la console AWS .
> Toutefois, si vous avez utilisé la fonction DLQ redrive sur la AWS console en août 2023, vous disposez d'une extension jusqu'au 15 octobre 2023 pour adopter les nouvelles autorisations selon l'une de ces options.
> Pour plus d'informations, consultez the section called “Identification des stratégies concernées”.

Vous trouverez ci-dessous des exemples d'autorisations de file d'attente pour chaque option de redirection de file d'attente de lettres mortes. Lorsque vous utilisez des files d'attente chiffrées côté serveur (SSE), l'autorisation AWS KMS clé correspondante est requise.

Basé sur l'action

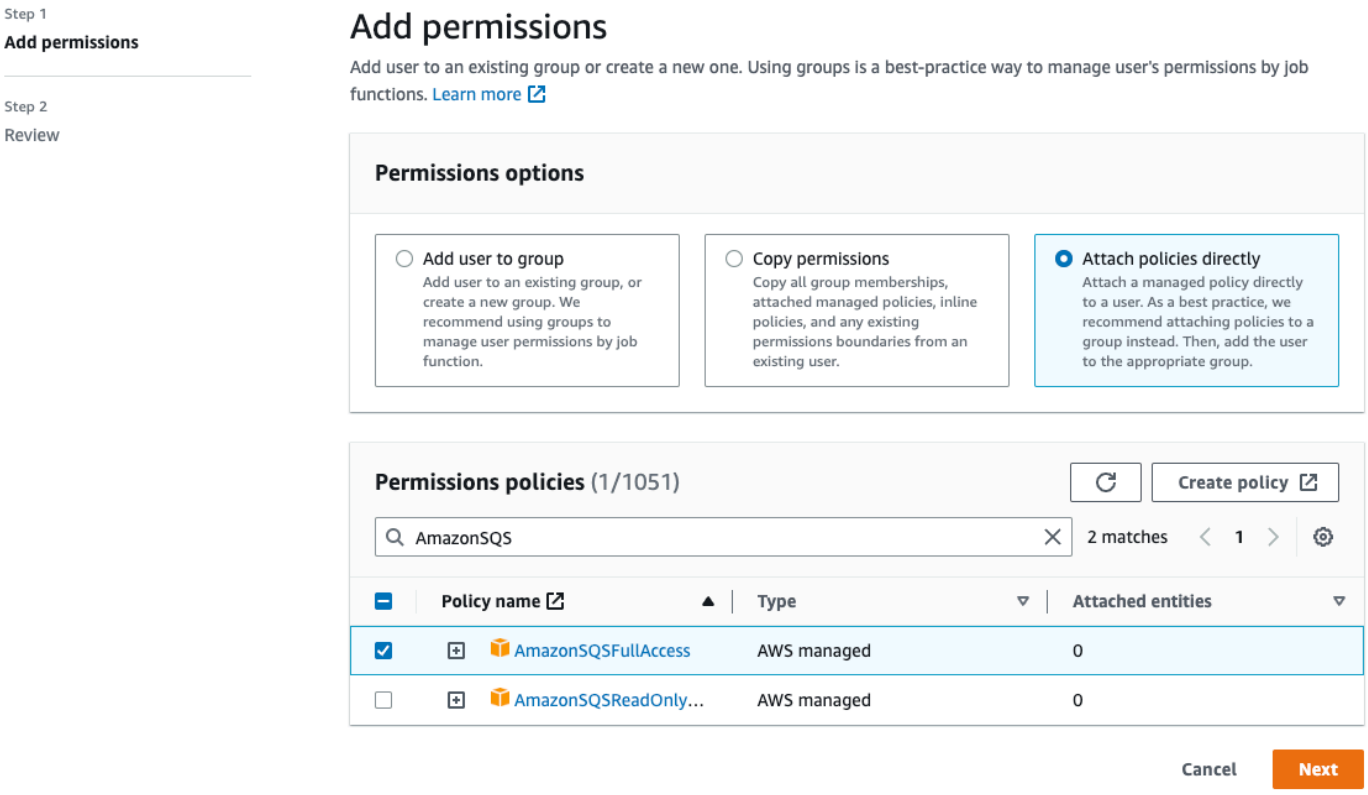
```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sqs:ReceiveMessage",
        "sqs:DeleteMessage",
        "sqs:GetQueueAttributes",
        "sqs:StartMessageMoveTask",
        "sqs:ListMessageMoveTasks",
        "sqs:CancelMessageMoveTask"
      ],
      "Resource": "arn:aws:sqs:<DLQ_region>:<DLQ_accountId>:<DLQ_name>"
    },
    {
      "Effect": "Allow",
      "Action": "sqs:SendMessage",
      "Resource":
 "arn:aws:sqs:<DestQueue_region>:<DestQueue_accountId>:<DestQueue_name>"
    }
  ]
}
```

Stratégie gérée

Les stratégies gérées suivantes contiennent les autorisations mises à jour requises :

- AmazonSQS FullAccess — Inclut les tâches de redrive de files d'attente en lettres mortes suivantes : démarrer, annuler et répertorier.
- Accès à AmazonSQS : fournit un ReadOnly accès en lecture seule et inclut la tâche de redrive de la liste des listes de lettres mortes.



Stratégie d'autorisation utilisant un caractère générique sqs*

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sqs:*",
      "Resource": "*"
    }
  ]
}
```

Identification des stratégies concernées

Si vous utilisez des politiques gérées par le client (CMP), vous pouvez utiliser AWS CloudTrail un IAM pour identifier les politiques affectées par la mise à jour des autorisations de file d'attente.

> **Note**
>
> Si vous utilisez `AmazonSQSFullAccess` et `AmazonSQSReadOnlyAccess`, aucune autre action n'est requise.

1. Connectez-vous à la AWS CloudTrail console.
2. Sur la page Historique des événements, sous Rechercher les attributs, utilisez le menu déroulant pour sélectionner le Nom de l'événement. Ensuite, recherchez `CreateMoveTask`.
3. Sélectionnez un événement pour ouvrir la page des Détails. Dans la section Enregistrements d'événements, récupérez le `UserName` ou le `RoleName` à partir de l'ARN `userIdentity`.
4. Connectez-vous à la console IAM.
   - Pour les utilisateurs, choisissez Utilisateurs. Sélectionnez l'utilisateur avec le `UserName` identifié à l'étape précédente.
   - Pour les rôles, choisissez Roles. Sélectionnez l'utilisateur avec le `RoleName` identifié à l'étape précédente.
5. Sur la page Détails, dans la section Autorisations, passez en revue les stratégies possédant le préfixe `sqs:` dans `Action`, ou consultez les stratégies pour lesquelles la file d'attente Amazon SQS est définie dans `Resource`.

Créez des alarmes pour les files d'attente de lettres mortes à l'aide d'Amazon CloudWatch

Vous pouvez configurer une alarme pour tous les messages placés dans une file d'attente de lettres mortes à l'aide d'Amazon CloudWatch et de la métrique. `ApproximateNumberOfMessagesVisible` Pour plus d'informations, consultez Création d' CloudWatch alarmes pour les métriques Amazon SQS. Une fois que vous avez reçu une alerte indiquant que des messages ont été envoyés dans la file d'attente des lettres mortes, vous pouvez consulter les messages à l'aide d'un sondage pour recevoir le message.

Métadonnées des messages pour Amazon SQS

Vous pouvez utiliser des attributs de message pour attacher des métadonnées personnalisées aux messages Amazon SQS de vos applications. Vous pouvez utiliser des attributs de système de messages pour stocker des métadonnées pour d'autres services AWS , tels que AWS X-Ray.

Rubriques

Attributs de message Amazon SQS

Amazon SQS vous permet d'inclure des métadonnées structurés (tels que des horodatages, des données géospatiales, des signatures et des identifiants) dans des messages utilisant des attributs de message. Chaque message peut contenir jusqu'à 10 attributs. Les attributs de message sont facultatifs et séparés du corps du message (même s'ils sont envoyés en même temps). Votre consommateur peut utiliser des attributs de message pour traiter un message d'une façon particulière sans avoir à traiter d'abord le corps du message. Pour plus d'informations sur l'envoi de messages avec des attributs à l'aide de la console Amazon SQS, consultez Envoi d'un message avec des attributs.

> **Note**
>
> Ne confondez pas les attributs des messages avec les attributs du système de messagerie : alors que vous pouvez utiliser les attributs de message pour joindre des métadonnées personnalisées aux messages Amazon SQS destinés à vos applications, vous pouvez utiliser les attributs du système de messagerie pour stocker les métadonnées d'autres AWS services, tels que. AWS X-Ray

Rubriques

Composants des attributs de message

> **Important**
>
> Tous les composants d'un attribut de message sont inclus dans la restriction de taille des messages à 256 Ko.

`Name`, `Type`, `Value` et le corps du message ne doivent pas être vides ni contenir la valeur null.

Chaque attribut de message est constitué des composants suivants :

- Nom : le nom de l'attribut de message peut contenir les caractères suivants : `A-Z`, `a-z`, `0-9`, trait de soulignement (_), tiret (`-`) et point (`.`). Les restrictions suivantes s'appliquent :
  - Il peut contenir jusqu'à 256 caractères.
  - Il ne peut pas commencer par `AWS.` ou `Amazon.` (ou toute variante de casse)
  - Il est sensible à la casse
  - Il doit être unique parmi tous les noms d'attribut pour le message
  - Il ne doit pas commencer ou se terminer par un point
  - Il ne doit pas comporter plusieurs points à la suite
- Type : type de données de l'attribut de message. Les types pris en charge incluent `String`, `Number` et `Binary`. Vous pouvez également ajouter des informations personnalisées pour tout type de données. Le type de données est soumis aux mêmes restrictions que le corps du message (pour plus d'informations, consultez `SendMessage` dans la Référence d'API Amazon Simple Queue Service). En outre, les restrictions suivantes s'appliquent :
  - Il peut contenir jusqu'à 256 caractères.
  - Il est sensible à la casse
- Valeur : valeur d'attribut du message. Pour les données de type `String`, les valeurs d'attribut sont soumises aux mêmes restrictions que le corps du message.

Types de données d'attribut de message

Les types de données d'attribut de message indiquent à Amazon SQS comment traiter les valeurs d'attribut de message correspondantes. Par exemple, si le type est `Number`, Amazon SQS valide les valeurs numériques.

Amazon SQS prend en charge les types de données logiques `String`, `Number` et `Binary`, avec la possibilité d'utiliser des étiquettes de type de données personnalisé au format *`.custom-data-type`*.

- Chaîne : les attributs `String` peuvent stocker n'importe quel texte Unicode à l'aide de caractères XML valides.

- Nombre : les attributs `Number` peuvent stocker des valeurs numériques positives ou négatives. Un nombre peut avoir jusqu'à 38 chiffres de précision et être compris entre 10^-128 et 10^+126.

> **Note**
>
> Amazon SQS supprime les zéros de début et de fin.

- Binaire : les binary attributes peuvent stocker n'importe quelle donnée binaire, telles que des données compressées, des données chiffrées ou des images.
- Personnalisé : pour créer un type de données personnalisé, ajoutez une étiquette de type personnalisé à n'importe quel type de données. Par exemple :
  - `Number.byte`, `Number.short`, `Number.int` et `Number.float` peuvent vous aider à faire la distinction entre les types de nombre.
  - `Binary.gif` et `Binary.png` peuvent vous aider à faire la distinction entre les types de fichier.

> **Note**
>
> Amazon SQS n'interprète pas, ne valide pas ou n'utilise pas les données ajoutées. L'étiquette de type personnalisé est soumise aux mêmes restrictions que le corps du message.

Calcul de la valeur de hachage MD5 pour les attributs de message

Si vous utilisez le AWS SDK for Java, vous pouvez ignorer cette section. La classe `MessageMD5ChecksumHandler` du kit SDK pour Java prend en charge les valeurs de hachage MD5 pour les attributs de message Amazon SQS.

Si vous utilisez l'API Query ou l'un des AWS SDK qui ne prennent pas en charge les résumés de messages MD5 pour les attributs de message Amazon SQS, vous devez suivre les directives suivantes pour effectuer le calcul du résumé des messages MD5.

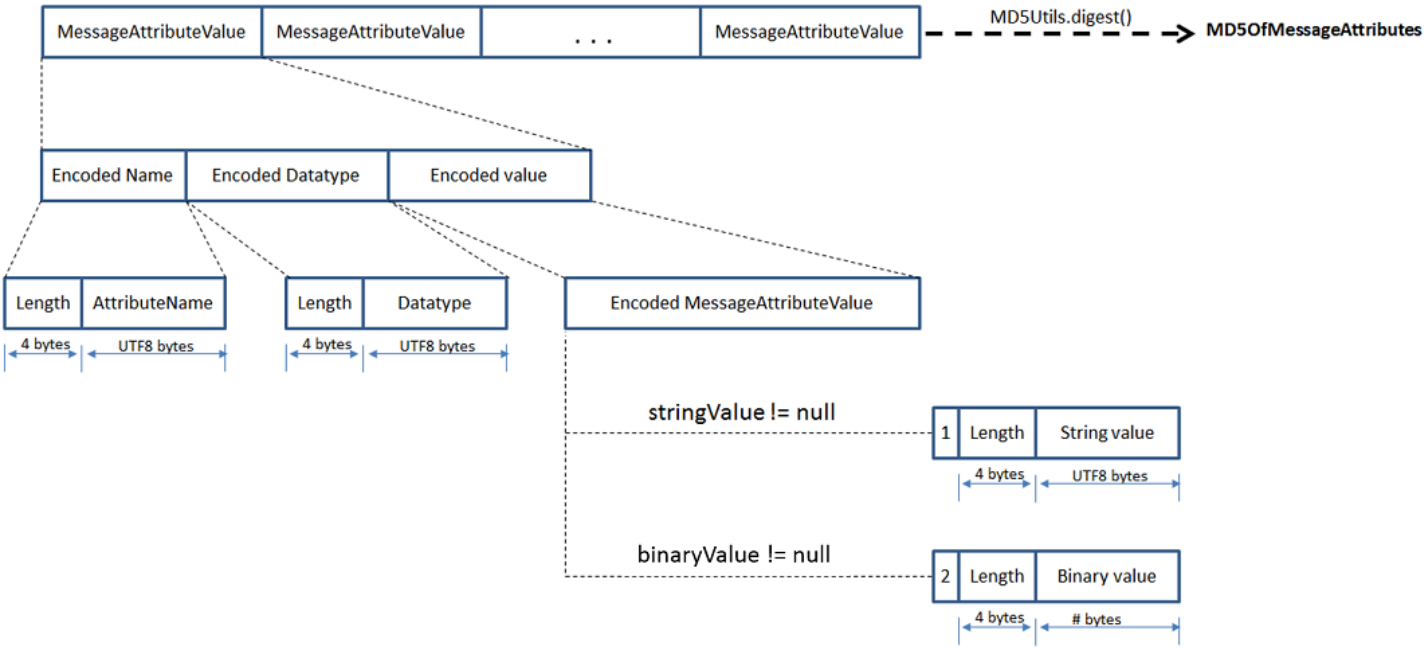
> **Note**
>
> Incluez toujours des suffixes de type de données personnalisés dans le calcul des valeurs de hachage MD5.

Présentation

Voici une présentation de l'algorithme de calcul de la valeur de hachage MD5 :

1. Triez tous les attributs de message dans l'ordre croissant en fonction de leur nom.
2. Encodez les différentes parties de chaque attribut (`Name`, `Type` et `Value`) dans un tampon.
3. Calculez le résumé du message du tampon entier.

Le schéma suivant illustre l'encodage du résumé de message MD5 pour un attribut de message unique :



Pour encoder un seul attribut de message Amazon SQS

1. Encodez le nom : la longueur (4 octets) et les octets UTF-8 du nom.
2. Encodez le type de données : la longueur (4 octets) et les octets UTF-8 du type de données.
3. Encodez le type de transport (`String` ou `Binary`) de la valeur (1 octet).

> **Note**
>
> Les types de données logiques `String` et `Number` utilisent le type de transport `String`. Le type de données logiques `Binary` utilise le type de transport `Binary`.

a. Pour le type de transport `String`, encodez 1.

b. Pour le type de transport `Binary`, encodez 2.

4. Encodez la valeur d'attribut.

a. Pour un type de transport `String`, encodez la valeur d'attribut : la longueur (4 octets) et les octets UTF-8 de la valeur.

b. Pour un type de transport `Binary`, encodez la valeur d'attribut : la longueur (4 octets) et les octets bruts de la valeur.

Attributs du système de message Amazon SQS

Vous pouvez utiliser des attributs de message pour attacher des métadonnées personnalisées aux messages Amazon SQS de vos applications, tandis que vous pouvez utiliser des attributs de système de message pour stocker des métadonnées pour d'autres services AWS , tels que AWS X-Ray. Pour plus d'informations, consultez le paramètre de demande `MessageSystemAttribute` des actions d'API `SendMessage` et `SendMessageBatch`, l'attribut `AWSTraceHeader` de l'action d'API `ReceiveMessage`, ainsi que le type de données `MessageSystemAttributeValue` dans la Référence d'API Amazon Simple Queue Service.

Les attributs de système de message sont structurés exactement comme les attributs de message, avec les exceptions suivantes :

- Actuellement, le seul attribut de système de message pris en charge est `AWSTraceHeader`. Son type `String` et sa valeur doivent être une chaîne d'en-tête de AWS X-Ray trace correctement formatée.
- La taille d'un attribut de système de message ne compte pas dans la taille totale d'un message.

Ressources requises pour traiter les messages Amazon SQS

Pour vous aider à estimer les ressources dont vous avez besoin pour traiter les messages en file d'attente, Amazon SQS peut déterminer le nombre approximatif de messages retardés, visibles et non visibles dans une file d'attente. Pour plus d'informations sur la visibilité, consultez la section Délai de visibilité Amazon SQS.

> **Note**
>
> Pour les files d'attente standard, le résultat est approximatif en raison de l'architecture distribuée d'Amazon SQS. Dans la plupart des cas, le nombre devrait être proche du nombre réel de messages dans la file d'attente.
> Pour les files d'attente FIFO, le résultat est exact.

Le tableau suivant indique le nom de l'attribut à utiliser avec l'action `GetQueueAttributes`.

Tâche	Nom d'attribut
Obtenir le nombre approximatif de messages disponibles dans la file d'attente.	`ApproximateNumberOfMessages Visible`
Obtenir le nombre approximatif de messages dans la file d'attente qui sont retardés et qui ne peuvent pas être lus immédiatement. Cela peut se produire lorsque la file d'attente est configurée avec un délai d'attente ou que le message a été envoyé avec un paramètre de délai d'attente.	`ApproximateNumberOfMessages Delayed`
Obtenir le nombre approximatif de messages en transit. Les messages sont considéré s comme en cours s'ils ont été expédiés à un client, mais qu'ils n'ont pas encore été supprimés ou qu'ils n'ont pas encore atteint la fin du délai de visibilité.	`ApproximateNumberOfMessages NotVisible`

Pagination des files d'attente

Les méthodes d'API `listQueues` et `listDeadLetterQueues` prennent en charge les contrôles de pagination facultatifs. Par défaut, ces méthodes d'API renvoient jusqu'à 1 000 files d'attente dans le message de réponse. Vous pouvez définir le paramètre `MaxResults` de manière à ce qu'il renvoie moins de résultats à chaque réponse.

Définissez le paramètre `MaxResults` dans la demande `listDeadLetterQueues` ou `listQueues` pour spécifier le nombre maximal de résultats à renvoyer dans la réponse. Si vous ne définissez pas `MaxResults`, la réponse inclut un maximum de 1 000 résultats et la valeur `NextToken` de la réponse est nulle.

Si vous définissez `MaxResults`, la réponse inclut une valeur pour `NextToken` s'il y a des résultats supplémentaires à afficher. Utilisez `NextToken` comme paramètre dans votre prochaine demande à `listQueues` pour recevoir la page de résultats suivante. La valeur `NextToken` de la réponse est nulle s'il n'y a pas de résultats supplémentaires à afficher.

Balises de répartition des coûts Amazon SQS

Pour organiser et identifier vos files d'attente Amazon SQS pour la répartition des coûts, vous pouvez ajouter des balises de métadonnées qui identifient le but, le propriétaire ou l'environnement d'une file d'attente. Cette approche est utile lorsque vous avez un grand nombre de files d'attente. Pour configurer les balises à l'aide de la console Amazon SQS, consultez the section called “Configuration de balises pour une file d'attente”

Vous pouvez utiliser des balises de répartition des coûts pour organiser votre AWS facture afin de refléter votre propre structure de coûts. Pour ce faire, inscrivez-vous pour que votre Compte AWS facture inclue les clés et les valeurs des tags. Pour plus d'informations, consultez Configuration du rapport de répartition des coûts mensuel dans le Guide d'utilisateur AWS Billing .

Chaque balise est composée d'une paire clé-valeur que vous définissez. Par exemple, vous pouvez facilement identifier vos files d'attente de production et de test en leur attribuant des balises comme suit :

File d'attente	Clé	Valeur
`MyQueueA`	`QueueType`	`Production`
`MyQueueB`	`QueueType`	`Testing`

> **Note**
>
> Lorsque vous utilisez des balises de file d'attente, tenez compte des consignes suivantes :

- Nous vous déconseillons d'ajouter plus de 50 balises à une file d'attente. Le balisage prend en charge les caractères Unicode en UTF-8.
- Les balises n'ont aucune signification sémantique. Amazon SQS interprète les balises en tant que chaîne de caractères.
- Les balises sont sensibles à la casse.
- Une nouvelle balise dont la clé est identique à celle d'une balise existante remplace la balise existante.
- Les actions de balisage sont limitées à 30 TPS par action. Compte AWS Si votre application nécessite un débit plus élevé, soumettez une demande.

Pour afficher la liste complète des restrictions de balise, consultez Quotas.

Recherches courtes et longues sur Amazon SQS

Amazon SQS propose des options d'interrogation courtes et longues pour recevoir des messages depuis une file d'attente. Tenez compte des exigences de réactivité et de rentabilité de votre application lorsque vous choisissez entre ces deux options de sondage :

- Interrogation courte (par défaut) : la `ReceiveMessage`demande interroge un sous-ensemble de serveurs (sur la base d'une distribution aléatoire pondérée) pour trouver les messages disponibles et envoie une réponse immédiate, même si aucun message n'est trouvé.
- Interrogation longue : `ReceiveMessage`interroge tous les serveurs à la recherche de messages, envoie une réponse dès qu'au moins un message est disponible, jusqu'au maximum spécifié. Une réponse vide n'est envoyée que si le temps d'attente pour le sondage expire. Cette option permet de réduire le nombre de réponses vides et potentiellement de réduire les coûts.

Les sections suivantes expliquent les détails des interrogations courtes et longues.

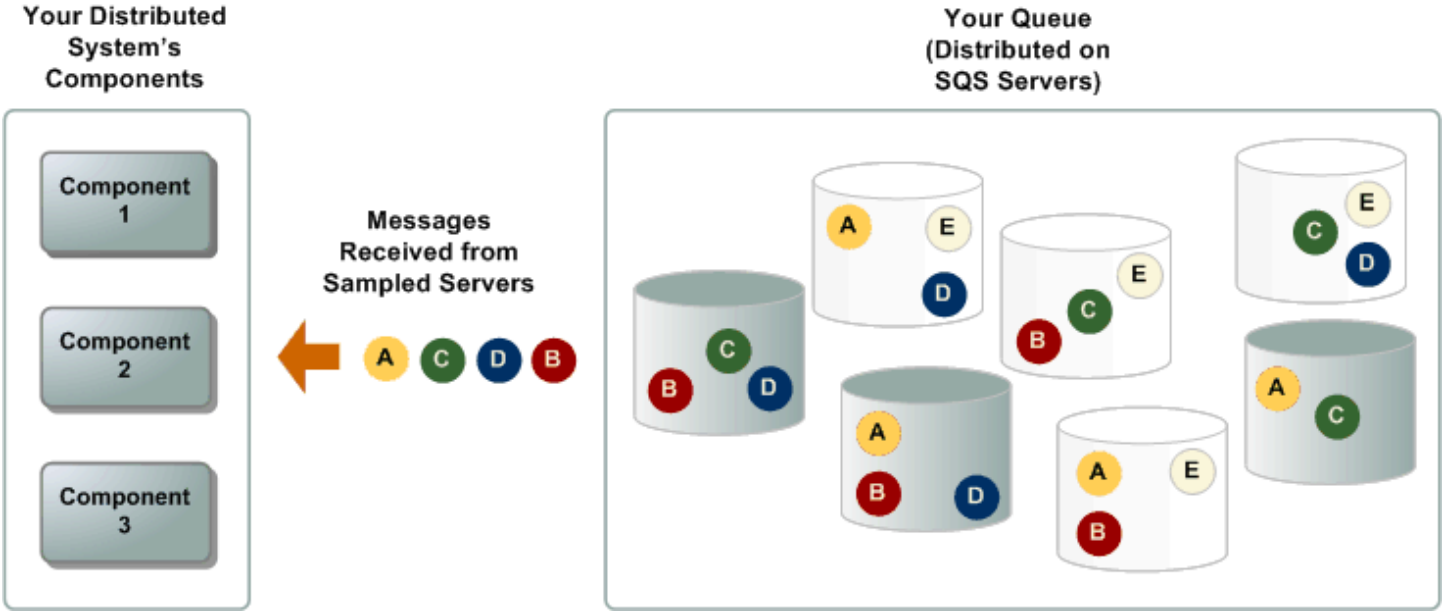
Rubriques

- Consommation des messages à l'aide de l'interrogation courte
- Consommation des messages à l'aide de la recherche prolongée
- Différences entre les interrogations courtes et longues

Consommation des messages à l'aide de l'interrogation courte

Lorsque vous consommez des messages d'une file d'attente (FIFO ou standard) à l'aide d'un court sondage, Amazon SQS échantillonne un sous-ensemble de ses serveurs (sur la base d'une distribution aléatoire pondérée) et renvoie des messages provenant uniquement de ces serveurs. Autrement dit, une demande `ReceiveMessage` particulière peut ne pas renvoyer tous les messages. Toutefois, si votre file d'attente compte moins de 1 000 messages, une requête ultérieure renvoie vos messages. Si vous continuez à consommer les messages de vos files d'attente, Amazon SQS sonde tous ses serveurs et vous recevez tous les messages.

Le schéma diagramme suivant illustre le comportement d'attente active de courte durée des messages renvoyés par une file d'attente standard suite à une demande réception de la part de l'un des composants du système. Amazon SQS sonde plusieurs de ses serveurs (en gris) et renvoie les messages A, C, D et B à partir de ces serveurs. Le message E n'est pas renvoyé pour cette requête, mais l'est pour une requête ultérieure.



Consommation des messages à l'aide de la recherche prolongée

Lorsque le temps d'attente pour l'action de l'API `ReceiveMessage` est supérieur à 0, une recherche prolongée est activée. Le temps d'attente maximal pour la recherche prolongée est de 20 secondes. La recherche prolongée permet de réduire le coût d'utilisation d'Amazon SQS en éliminant le nombre de réponses vides (lorsqu'il n'y a aucun message disponible pour une demande `ReceiveMessage`) et de fausses réponses vides (lorsque les messages sont disponibles dans la file d'attente, mais ne sont pas inclus dans une réponse). Pour plus d'informations sur l'activation d'une recherche

prolongée pour une file d'attente nouvelle ou existante à l'aide de la console Amazon SQS, consultez Configuration des paramètres de file d'attente à l'aide de la console Amazon SQS. Pour connaître les bonnes pratiques, consultez Configuration de l'interrogation longue.

L'attente active de longue durée offre les avantages suivants :

- Réduisez les réponses vides en permettant à Amazon SQS d'attendre qu'un message soit disponible dans une file d'attente avant d'envoyer une réponse. A moins que la connexion expire, la réponse à la demande `ReceiveMessage` contient au moins l'un des messages disponibles et, au plus, le nombre maximum de messages spécifiés dans l'action `ReceiveMessage`. Dans de rares cas, vous pouvez recevoir des réponses vides même si une file d'attente contient encore des messages, en particulier si vous spécifiez une faible valeur pour le paramètre `ReceiveMessageWaitTimeSeconds`.
- Réduisez les fausses réponses vides en interrogeant tous les serveurs Amazon SQS, plutôt qu'un sous-ensemble de ceux-ci.
- Renvoyez des messages dès qu'ils sont disponibles.

Pour obtenir des informations sur la façon de vérifier qu'une file d'attente est vide, consultez Confirmation qu'une file d'attente Amazon SQS est vide.

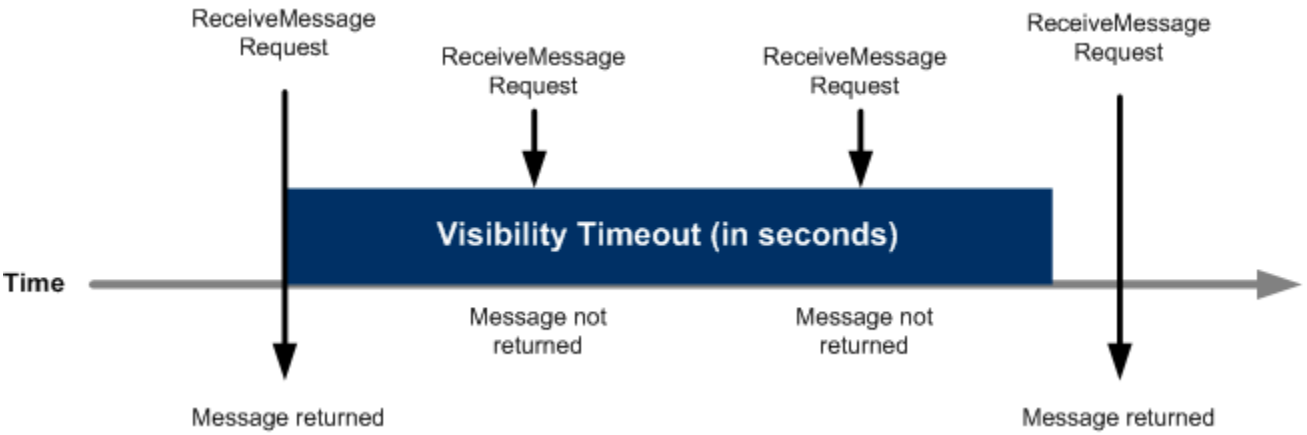
Différences entre les interrogations courtes et longues

L'attente active de courte durée survient lorsque le paramètre `WaitTimeSeconds` d'une réponse `ReceiveMessage` est défini sur `0` de l'une des deux manières suivantes :

- L'appel de `ReceiveMessage` définit `WaitTimeSeconds` sur `0`.
- L'appel `ReceiveMessage` ne définit pas `WaitTimeSeconds`, mais l'attribut de file d'attente `ReceiveMessageWaitTimeSeconds` est défini sur `0`.

Délai de visibilité Amazon SQS

Lorsqu'un client reçoit et traite un message à partir d'une file d'attente, le message reste dans celle-ci. Amazon SQS ne supprime pas automatiquement le message. Dans la mesure où Amazon SQS est un système distribué, rien ne garantit que le consommateur recevra réellement le message (par exemple, en cas de problème de connectivité ou de problème de l'application du consommateur). L'utilisateur doit donc supprimer le message de la file d'attente après l'avoir reçu et traité.



Juste après réception d'un message, il reste dans la file d'attente. Afin d'empêcher les autres consommateurs de traiter le message à nouveau, Amazon SQS définit un délai de visibilité, une période au cours de laquelle Amazon SQS empêche d'autres consommateurs de recevoir et de traiter le message. Le délai de visibilité par défaut d'un message est de 30 secondes. La valeur minimale est 0 seconde. La valeur maximale est 12 heures. Pour plus d'informations sur la configuration du délai de visibilité pour une file d'attente à l'aide de la console, consultez Configuration des paramètres de file d'attente à l'aide de la console Amazon SQS.

> **Note**
>
> Pour les files d'attente standard, le délai de visibilité n'est pas une garantie contre la réception d'un message en double. Pour plus d'informations, consultez Une t-least-once livraison.
> Les files d'attente FIFO permettent au producteur ou au consommateur d'effectuer plusieurs tentatives :
>
> - Si le producteur détecte l'échec d'une action `SendMessage`, il peut réessayer d'en envoyer autant de fois que nécessaire, en utilisant le même identifiant de déduplication des messages. En supposant que le producteur reçoive au moins un accusé de réception avant l'expiration de l'intervalle de déduplication, les tentatives multiples n'affectent pas l'ordre des messages et n'introduisent pas de doublons.
> - Si le consommateur détecte l'échec d'une action `ReceiveMessage`, il peut réessayer autant de fois que nécessaire, en utilisant le même identifiant de tentative de demande de réception. En supposant que le consommateur reçoive au moins un accusé de réception avant l'expiration du délai de visibilité, les tentatives multiples n'ont aucune incidence sur l'ordre des messages.

- Lorsque vous recevez un message avec un ID de groupe de messages, aucun autre message correspondant au même ID de groupe de messages n'est renvoyé, sauf si vous supprimez le message ou s'il devient visible.

Rubriques

Messages en cours

Un message Amazon SQS possède trois états de base :

1. Envoyé vers une file d'attente par un producteur.
2. Reçu de la file d'attente par un consommateur.
3. Supprimé de la file d'attente.

Un message est considéré comme étant stocké après qu'il a été envoyé à une file d'attente par un producteur, mais qu'il n'a pas encore été reçu depuis la file d'attente par un consommateur (c'est-à-dire entre les états 1 et 2). Il n'y a pas de quota quant au nombre de messages enregistrés. Un message est considéré comme étant en cours après qu'il a été reçu depuis une file d'attente par un consommateur, mais pas encore supprimé de la file d'attente (c'est-à-dire entre les états 2 et 3). Il existe un quota quant au nombre de messages en cours.

Important

Les quotas qui s'appliquent aux messages en cours ne sont pas liés au nombre illimité de messages stockés.

Pour la plupart des files d'attente standard (en fonction du trafic de la file d'attente et du backlog de messages), il peut y avoir un maximum d'environ 120 000 messages en cours (reçus depuis une file d'attente par un consommateur, mais pas encore supprimés de la file d'attente). Si vous atteignez ce quota tout en utilisant la recherche courte, Amazon SQS renvoie le message d'erreur `OverLimit`.

Si vous utilisez la recherche prolongée, Amazon SQS ne renvoie aucun message d'erreur. Pour éviter d'atteindre cette limite, supprimez les messages de la file d'attente une fois qu'ils ont été traités. Vous pouvez également augmenter le nombre de files d'attente que vous utilisez pour traiter vos messages. Pour demander une augmentation de quota, envoyez une demande de support.

Pour les files d'attente FIFO, il peut y avoir un maximum de 20 000 messages en cours (reçus depuis une file d'attente par un consommateur, mais pas encore supprimés de la file d'attente). Si vous atteignez ce quota, Amazon SQS ne renvoie aucun message d'erreur.

> **Important**
>
> Lorsque vous travaillez avec des files d'attente FIFO, les opérations `DeleteMessage` échoueront si la demande est reçue en dehors du délai de visibilité. Si le délai de visibilité est de 0 seconde, le message doit être supprimé dans la milliseconde dans laquelle il a été envoyé, sinon, il est considéré comme abandonné. Cela peut amener Amazon SQS à inclure des messages en double dans la même réponse à une opération `ReceiveMessage` si le paramètre `MaxNumberOfMessages` est supérieur à 1. Pour plus de détails, consultez Comment fonctionne l'API FIFO Amazon SQS.

Définition du délai de visibilité

Le délai de visibilité commence lorsque Amazon SQS renvoie un message. Pendant ce temps, le consommateur traite et supprime le message. Toutefois, si le consommateur rencontre un échec avant de supprimer le message et que votre système n'appelle pas l'action `DeleteMessage` pour ce message avant l'expiration du délai de visibilité, le message devient visible pour les autres consommateurs et est reçu à nouveau. Si un message ne doit être reçu qu'une seule fois, votre consommateur doit le supprimer avant l'expiration du délai de visibilité.

Pour chaque file d'attente Amazon SQS, le délai de visibilité par défaut est de 30 secondes. Vous pouvez modifier ce paramètre pour toute la file d'attente. Généralement, vous devez configurer le délai de visibilité en fonction du temps maximal nécessaire à votre application pour traiter et supprimer un message de la file d'attente. Lors de la réception des messages, vous pouvez également définir un délai de visibilité qui leur est spécifique, sans modifier le délai de visibilité de toute la file d'attente. Pour en savoir plus, consultez les bonnes pratiques dans la section Traitement des messages en temps opportun.

Si vous ne savez pas combien de temps il faut pour traiter un message, créez une pulsation pour votre processus de consommateur : spécifiez le délai de visibilité initial (par exemple, 2 minutes) puis,

tant que votre client travaille sur le message, continuez à prolonger le délai de visibilité de 2 minutes, toutes les minutes.

> **Important**
>
> Le délai maximal de visibilité est de 12 heures à compter de l'heure où Amazon SQS reçoit la demande `ReceiveMessage`. L'extension du délai d'attente de visibilité ne réinitialise pas le maximum de 12 heures.
> En outre, il se peut que vous ne puissiez pas régler le délai d'expiration d'un message individuel sur 12 heures (par exemple, 43 200 secondes) puisque la demande `ReceiveMessage` déclenche le temporisateur. Par exemple, si vous recevez un message et que vous définissez immédiatement le maximum de 12 heures en envoyant un appel `ChangeMessageVisibility` avec `VisibilityTimeout` défini sur une durée égale à 43 200 secondes, il échouera probablement. En revanche, l'utilisation d'une valeur de 43 195 secondes fonctionnera, à moins qu'il n'y ait un délai important entre la demande du message via `ReceiveMessage` et la mise à jour du délai de visibilité. Si votre client a besoin de plus de 12 heures, envisagez d'utiliser Step Functions.

Modification du délai de visibilité d'un message

Lorsque vous recevez un message provenant d'une file d'attente et que vous commencez à le traiter, le délai de visibilité de cette dernière peut être insuffisant (par exemple, vous pouvez avoir besoin de traiter et supprimer un message). Pour raccourcir ou rallonger le délai de visibilité d'un message, spécifiez une nouvelle valeur à l'aide de l'action `ChangeMessageVisibility`.

Par exemple, si le délai d'expiration par défaut pour une file d'attente est de 60 secondes, que 15 secondes se sont écoulées depuis que vous avez reçu le message, et que vous envoyez un appel `ChangeMessageVisibility` avec `VisibilityTimeout` défini sur 10 secondes, les 10 secondes commençant à partir du moment que vous effectuez l'appel `ChangeMessageVisibility`. Par conséquent, toute tentative de modifier le délai de visibilité ou de supprimer ce message 10 secondes après que vous avez initialement modifié le délai de visibilité (un total de 25 secondes) peut entraîner une erreur.

> **Note**
>
> Le nouveau délai de visibilité prend effet à partir du moment où vous appelez l'action `ChangeMessageVisibility`. De plus, le nouveau délai s'applique uniquement à la

réception spécifique de ce message. `ChangeMessageVisibility` n'a pas d'incidence sur le délai de visibilité des réceptions ou des files d'attente ultérieures.

Désactivation du délai de visibilité d'un message

Lorsque vous recevez un message à partir d'une file d'attente, vous ne souhaitez pas toujours le traiter et le supprimer. Amazon SQS vous permet de mettre fin au délai de visibilité pour un message spécifique. Dans ce cas, les autres composants du système voient immédiatement le message et peuvent le traiter.

Pour désactiver le délai de visibilité d'un message après avoir appelé `ReceiveMessage`, appelez `ChangeMessageVisibility` en définissant `VisibilityTimeout` sur 0 seconde.

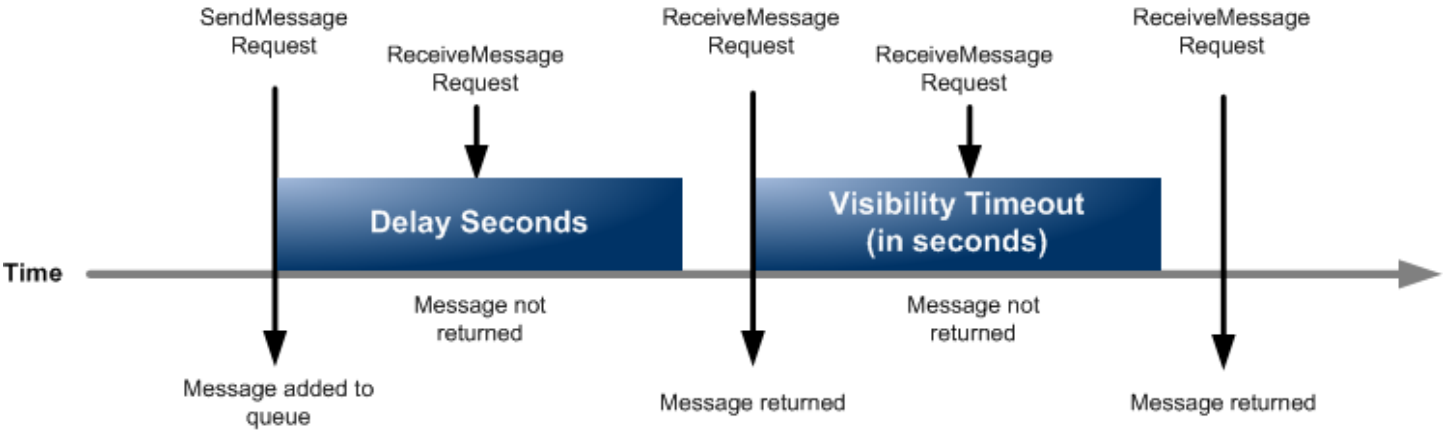
Files d'attente à retardement Amazon SQS

Les files d'attente à retardement vous permettent de reporter la livraison des nouveaux messages aux clients de quelques secondes, par exemple lorsque votre application client a besoin de plus de temps pour traiter les messages. Si vous créez une file d'attente à retardement, les consommateurs ne peuvent pas voir les messages que vous envoyez à la file d'attente pendant toute la durée du retardement. Le délai (minimum) par défaut pour une file d'attente est de 0 seconde. La valeur maximale est de 15 minutes. Pour plus d'informations sur la configuration de files d'attente à retardement à l'aide de la console, consultez Configuration des paramètres de file d'attente à l'aide de la console Amazon SQS.

> **Note**
>
> Pour les files d'attente standard, le retard par file d'attente n'est pas rétroactif : la modification du paramètre n'affecte pas le retard des messages déjà présents dans la file d'attente.
> Pour les files d'attente FIFO, le retard par file d'attente est rétroactif : la modification du paramètre affecte le retard des messages déjà présents dans la file d'attente.

Les files d'attente à retardement sont similaires aux délais de visibilité, car ces deux fonctions empêchent les utilisateurs d'accéder aux messages pendant une période donnée. La différence entre les deux est que pour, les files d'attente à retardement, un message est masqué lorsqu'il est ajouté initialement à la file d'attente, tandis que pour les délais de visibilité, un message est masqué

uniquement après sa consommation de la file d'attente. Le diagramme suivant illustre la relation entre les files d'attente à retardement et les délais de visibilité.



Pour définir un retard en secondes pour des messages individuels, plutôt que pour une file d'attente complète, utilisez des temporisateurs de message pour autoriser Amazon SQS à utiliser la valeur `DelaySeconds` du temporisateur de messages au lieu de la valeur `DelaySeconds` de la file d'attente à retardement.

Files d'attente temporaires Amazon SQS

Les files d'attente temporaires vous permettent de gagner du temps de développement et de réduire les coûts de déploiement lorsque vous utilisez des modèles de messages courants tels que demande-réponse. Vous pouvez utiliser le Client de file d'attente temporaire pour créer des files d'attente temporaires à haut débit, économiques et gérées par l'application.

Le client mappe automatiquement plusieurs files d'attente temporaires (des files d'attente gérées par des applications créées à la demande pour un processus particulier) sur une seule file d'attente Amazon SQS. Cela permet à votre application d'effectuer moins d'appels d'API et de bénéficier d'un meilleur débit lorsque le trafic vers chaque file d'attente temporaire est faible. Lorsqu'une file d'attente temporaire n'est plus utilisée, le client la supprime automatiquement, même si certains processus qui utilisent le client ne sont pas fermés correctement.

Voici les avantages des files d'attente temporaires :

- Celles-ci servent des canaux de communication légers pour des threads ou processus spécifiques.
- Elles peuvent être créées et supprimées sans encourir de frais supplémentaires.

- Elles sont compatibles API avec des files d'attente Amazon SQS statiques (normales). Cela signifie que le code existant qui envoie et reçoit des messages peut envoyer des messages et recevoir des messages dans des files d'attente virtuelles.

Rubriques

Files d'attente virtuelles

Les files d'attente virtuelles sont des structures de données locales créées par le client de file d'attente temporaire. Les files d'attente virtuelles vous permettent de combiner plusieurs destinations à faible trafic en une seule file d'attente Amazon SQS. Pour connaître les bonnes pratiques, consultez Évitez de réutiliser le même ID de groupe de messages avec des files d'attente virtuelles.

> **Note**
>
> - La création d'une file d'attente virtuelle crée uniquement des structures de données temporaires dans lesquelles les consommateurs reçoivent les messages. Comme une file d'attente virtuelle n'effectue aucun appel d'API vers Amazon SQS, les files d'attente virtuelles n'entraînent aucun coût.
> - Les quotas TPS s'appliquent à toutes les files d'attente virtuelles sur une file d'attente hôte unique. Pour plus d'informations, consultez Quotas de messages Amazon SQS.

La classe wrapper `AmazonSQSVirtualQueuesClient` ajoute la prise en charge des attributs liés aux files d'attente virtuelles. Pour créer une file d'attente virtuelle, vous devez appeler l'action d'API `CreateQueue` à l'aide de l'attribut `HostQueueURL`. Cet attribut spécifie la file d'attente existante qui héberge les files d'attente virtuelles.

L'URL d'une file d'attente virtuelle est au format suivant.

```
https://sqs.us-east-2.amazonaws.com/123456789012/MyQueue#MyVirtualQueueName
```

Lorsqu'un producteur appelle l'action d'API `SendMessage` ou `SendMessageBatch` sur une URL de file d'attente virtuelle, le client de file d'attente temporaire effectue les opérations suivantes :

1. Il extrait le nom de file d'attente virtuelle.
2. Il attache un nom de file d'attente virtuelle en tant qu'attribut de message supplémentaire.
3. Il envoie le message dans la file d'attente d'hôte.

Pendant que le producteur envoie des messages, un thread d'arrière-plan interroge la file d'attente hôte et envoie les messages reçus vers des files d'attente virtuelles en fonction des attributs de message correspondants.

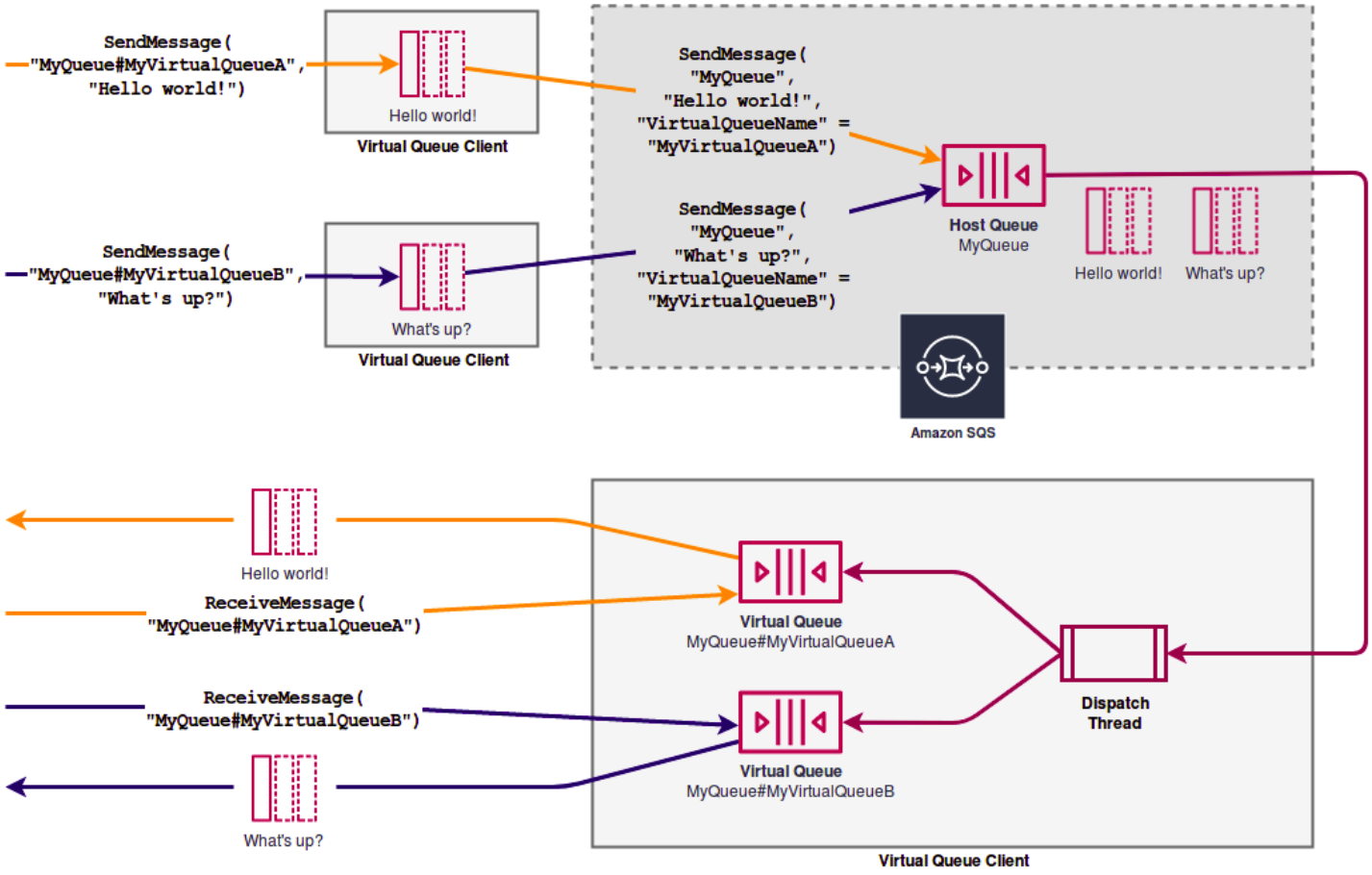
Pendant que le consommateur appelle l'action d'API `ReceiveMessage` sur une URL de file d'attente virtuelle, le client de file d'attente temporaire bloque l'appel localement jusqu'à ce que le thread d'arrière-plan envoie un message dans la file d'attente virtuelle. (Ce processus est similaire à la récupération préalable des messages dans le client asynchrone mis en tampon : une seule action d'API peut fournir jusqu'à 10 messages à des files d'attente virtuelles.) La suppression d'une file d'attente virtuelle supprime les ressources côté client sans appeler Amazon SQS proprement dit.

La classe `AmazonSQSTemporaryQueuesClient` transforme automatiquement toutes les files d'attente qu'elle crée en files d'attente temporaires. Elle crée également automatiquement des files d'attente hôte avec les mêmes attributs de file d'attente, à la demande. Ces noms de files d'attente partagent un préfixe configurable commun (par défaut, `__RequesterClientQueues__`) qui les identifie comme des files d'attente temporaires. Cela permet au client d'agir comme un remplacement optimisant le code existant qui crée et supprime les files d'attente. Le client inclut également les interfaces `AmazonSQSRequester` et `AmazonSQSResponder` qui permettent la communication bidirectionnelle entre les files d'attente.

Modèle de messagerie demande-réponse (files d'attente virtuelles)

Le cas d'utilisation le plus courant pour les files d'attente temporaires est le modèle de messagerie demande-réponse dans lequel un demandeur crée une file d'attente temporaire pour la réception de chaque message de réponse. Pour éviter la création d'une file d'attente Amazon SQS pour chaque message de réponse, le client de file d'attente temporaire vous permet de créer et de supprimer plusieurs files d'attente temporaires sans effectuer des appels d'API Amazon SQS. Pour plus d’informations, consultez Implémentation de systèmes de demande-réponse.

Le schéma suivant montre une configuration commune qui utilise ce modèle.



Exemple de scénario : Traitement d'une demande de connexion

L'exemple de scénario suivant montre comment utiliser les interfaces `AmazonSQSRequester` et `AmazonSQSResponder` pour traiter une demande de connexion d'un utilisateur.

Côté client

```
public class LoginClient {

    // Specify the Amazon SQS queue to which to send requests.
    private final String requestQueueUrl;

    // Use the AmazonSQSRequester interface to create
    // a temporary queue for each response.
    private final AmazonSQSRequester sqsRequester =
            AmazonSQSRequesterClientBuilder.defaultClient();

    LoginClient(String requestQueueUrl) {
```

```
        this.requestQueueUrl = requestQueueUrl;
    }

    // Send a login request.
    public String login(String body) throws TimeoutException {
        SendMessageRequest request = new SendMessageRequest()
                .withMessageBody(body)
                .withQueueUrl(requestQueueUrl);

        // If no response is received, in 20 seconds,
        // trigger the TimeoutException.
        Message reply = sqsRequester.sendMessageAndGetResponse(request,
                20, TimeUnit.SECONDS);

        return reply.getBody();
    }
}
```

L'envoi d'une demande de connexion effectue les opérations suivantes :

1. Crée une table temporaire.
2. Attache l'URL de la file d'attente temporaire au message en tant qu'attribut.
3. Envoie le message.
4. Reçoit une réponse de la file d'attente temporaire.
5. Supprime la file d'attente temporaire.
6. Renvoie la réponse.

Côté serveur

L'exemple suivant suppose que, lors de la construction, un thread est créé pour interroger la file d'attente et appeler la méthode `handleLoginRequest()` pour chaque message. En outre, `doLogin()` est une méthode assumée.

```
public class LoginServer {

    // Specify the Amazon SQS queue to poll for login requests.
    private final String requestQueueUrl;

    // Use the AmazonSQSResponder interface to take care
    // of sending responses to the correct response destination.
```

```
    private final AmazonSQSResponder sqsResponder =
            AmazonSQSResponderClientBuilder.defaultClient();

    LoginServer(String requestQueueUrl) {
        this.requestQueueUrl = requestQueueUrl;
    }

    // Process login requests from the client.
    public void handleLoginRequest(Message message) {

        // Process the login and return a serialized result.
        String response = doLogin(message.getBody());

        // Extract the URL of the temporary queue from the message attribute
        // and send the response to the temporary queue.
        sqsResponder.sendResponseMessage(MessageContent.fromMessage(message),
                new MessageContent(response));
    }
}
```

Nettoyage des files d'attente

Pour faire en sorte qu'Amazon SQS récupère les ressources en mémoire utilisées par les files d'attente virtuelles, lorsque votre application n'a plus besoin du client de file d'attente temporaire, il doit appeler la méthode `shutdown()`. Vous pouvez également utiliser la méthode `shutdown()` de l'interface `AmazonSQSRequester`.

Le client de file d'attente temporaire fournit également un moyen d'éliminer les files d'attente hôte orphelines. Pour chaque file d'attente qui reçoit un appel d'API sur une période de temps (par défaut, cinq minutes), le client utilise l'action d'API `TagQueue` pour ajouter des balises à une file d'attente qui continue d'être utilisée.

> **Note**
>
> Toute action d'API effectuée sur une file d'attente marque celle-ci comme étant non inactive, y compris une action `ReceiveMessage` qui ne renvoie aucun message.

Le thread d'arrière-plan utilise les actions d'API `ListQueues` et `ListTags` pour vérifier toutes les files d'attente avec le préfixe configuré, en supprimant les files d'attente qui n'ont pas été balisées pendant au moins cinq minutes. Ainsi, si un client n'est pas fermé correctement, les autres clients

actifs sont nettoyés après celui-ci. Afin de réduire la duplication de travail, tous les clients avec le même préfixe communiquent via une file d'attente de travail interne partagée, nommée d'après le préfixe.

Temporisateurs de messages Amazon SQS

Les temporisateurs de messages vous permettent de définir une période d'invisibilité initiale pour un message ajouté à une file d'attente. Par exemple, si vous envoyez un message avec un temporisateur de 45 secondes, le message n'est pas visible pour les consommateurs pendant ses 45 premières secondes dans la file d'attente. Le délai (minimum) par défaut pour un message est de 0 seconde. La valeur maximale est de 15 minutes. Pour plus d'informations sur l'envoi de messages avec des temporisateurs à l'aide de la console, consultez Envoyer un message.

> **Note**
>
> Les files d'attente FIFO ne prennent pas en charge les temporisateurs pour les messages individuels.

Pour définir une période de retard pour une file d'attente complète plutôt qu'au niveau de messages individuels, utilisez des files d'attente à retardement. Les valeurs des temporisateurs définis au niveau des messages individuels prévalent sur les valeurs `DelaySeconds` d'une file d'attente à retardement Amazon SQS.

Accès à Amazon EventBridge Pipes via la console Amazon SQS

Amazon EventBridge Pipes connecte les sources aux cibles. Les tubes sont destinés aux point-to-point intégrations entre les sources et les cibles prises en charge, avec la prise en charge des transformations avancées et de l'enrichissement. EventBridge Les pipes constituent un moyen hautement évolutif de connecter votre file d'attente Amazon SQS à des AWS services tels que Step Functions, Amazon SQS et API Gateway, ainsi qu'à des applications logicielles en tant que service (SaaS) tierces telles que Salesforce.

Pour configurer un canal, vous devez choisir la source, ajouter un filtrage facultatif, définir un enrichissement facultatif et choisir la cible pour les données d'événement.

Sur la page de détails d'une file d'attente Amazon SQS, vous pouvez voir les pipelines qui utilisent cette file d'attente comme source. À partir de là, vous pouvez également :

- Lancez la EventBridge console pour afficher les détails du canal.
- Lancez la EventBridge console pour créer un nouveau canal avec la file d'attente comme source.

Pour plus d'informations sur la configuration d'une file d'attente Amazon SQS en tant que source de canal, consultez la section file d'attente Amazon SQS en tant que source dans le guide de l'utilisateur Amazon EventBridge . Pour plus d'informations sur EventBridge les tuyaux en général, voir EventBridge Tuyaux.

Pour accéder aux EventBridge canaux d'une file d'attente Amazon SQS donnée

1. Ouvrez la page Files d'attente de la console Amazon SQS.
2. Sélectionnez une file d'attente.
3. Sur la page détaillée de la file d'attente, choisissez l'onglet EventBridge Canalisations.

   L'onglet EventBridge Canalisations inclut une liste de tous les canaux actuellement configurés pour utiliser la file d'attente sélectionnée comme source, notamment :

   - nom du canal
   - état actuel
   - cible du canal
   - date à laquelle le canal a été modifié pour la dernière fois
4. Affichez plus de détails sur le pipeline ou créez-en un, si vous le souhaitez :

   - Pour accéder à plus de détails sur un pipeline :

     Sélectionnez le nom du pipeline.

     Cela ouvre la page de détails de Pipe de la EventBridge console.
   - Pour créer un pipeline :

     Choisissez Connecter la file d'attente Amazon SQS au pipeline.

     Cela lance la page Créer un canal de la EventBridge console, avec la file d'attente Amazon SQS spécifiée comme source de canal. Pour plus d'informations, consultez la section Création d'un EventBridge canal dans le guide de EventBridge l'utilisateur Amazon.

> **Important**
>
> Un message d'une file d'attente Amazon SQS est lu par un seul pipeline, puis supprimé de la file d'attente après avoir été traité, que le message corresponde ou non au filtre que vous avez configuré pour ce pipeline. Procédez avec prudence lorsque vous configurez plusieurs pipelines pour utiliser la même file d'attente que la source.

Gestion de messages Amazon SQS volumineux avec Extended Client Library et Amazon Simple Storage Service

Vous pouvez utiliser la bibliothèque client étendue Amazon SQS pour Java et la bibliothèque client étendue Amazon SQS pour Python pour envoyer des messages volumineux. Cela est particulièrement utile pour consommer des charges utiles de messages volumineuses, de 256 Ko à 2 Go. Les deux bibliothèques enregistrent la charge utile du message dans un compartiment Amazon Simple Storage Service et envoient la référence de l'objet Amazon S3 stocké à la file d'attente Amazon SQS.

> **Note**
>
> Les bibliothèques client étendues Amazon SQS sont compatibles avec les files d'attente standard et FIFO.

Rubriques

- Gestion de messages Amazon SQS volumineux à l'aide de Java et Amazon S3
- Gestion de messages Amazon SQS volumineux à l'aide de Python et Amazon S3

Gestion de messages Amazon SQS volumineux à l'aide de Java et Amazon S3

Vous pouvez utiliser la bibliothèque client étendue Amazon SQS pour Java et Amazon Simple Storage Service (Amazon S3) pour gérer les messages Amazon Simple Queue Service (Amazon SQS) volumineux. Cela est particulièrement utile pour consommer des charges utiles de messages volumineuses, allant de 256 Ko à 2 Go. La bibliothèque enregistre la charge utile du message dans

un compartiment Amazon S3 et envoie un message contenant une référence à l'objet Amazon S3 stocké à une file d'attente Amazon SQS.

Vous pouvez utiliser la bibliothèque client étendue Amazon SQS pour Java pour effectuer les opérations suivantes :

- Spécifier si les messages sont toujours stockés dans Amazon S3 ou seulement lorsque leur taille dépasse 256 Ko.
- Envoyer un message qui fait référence un objet de message unique stocké dans un compartiment S3.
- Récupérez l'objet du message depuis un compartiment Amazon S3
- Supprimer l'objet du message d'un compartiment Amazon S3

Prérequis

L'exemple suivant utilise le SDK AWS Java. Pour installer et configurer le SDK, consultez la section Configurer le AWS SDK pour Java dans le Guide AWS SDK for Java du développeur.

Avant d'exécuter l'exemple de code, configurez vos AWS informations d'identification. Pour plus d'informations, consultez la section Configurer les AWS informations d'identification et la région pour le développement dans le guide du AWS SDK for Java développeur.

Le kit SDK pour Java et la bibliothèque client étendue Amazon SQS pour Java nécessitent le kit de développement J2SE 8.0 ou version ultérieure.

> **Note**
>
> Vous pouvez utiliser la bibliothèque client étendue Amazon SQS pour Java afin de gérer les messages Amazon SQS à l'aide d'Amazon S3 uniquement avec le AWS SDK for Java. Vous ne pouvez pas le AWS CLI faire avec la console Amazon SQS, l'API HTTP Amazon SQS ou tout autre SDK. AWS

AWS Exemple de SDK pour Java 2.x : utilisation d'Amazon S3 pour gérer des messages Amazon SQS volumineux

L'exemple de AWS SDK pour Java 2.x suivant crée un compartiment Amazon S3 avec un nom aléatoire et ajoute une règle de cycle de vie pour supprimer définitivement les objets après 14 jours.

Il crée également une file d'attente nommée `MyQueue` et envoie un message aléatoire qui est stocké dans un compartiment S3 et dont la taille est supérieure à 256 Ko pour la file d'attente. Enfin, le code récupère le message, renvoie des informations sur ce dernier et le supprime, ainsi que la file d'attente et le compartiment.

```
/*
 * Copyright 2010-2024 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

import com.amazon.sqs.javamessaging.AmazonSQSExtendedClient;
import com.amazon.sqs.javamessaging.ExtendedClientConfiguration;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.*;
import com.amazonaws.services.sqs.AmazonSQS;
import com.amazonaws.services.sqs.AmazonSQSClientBuilder;
import com.amazonaws.services.sqs.model.*;
import org.joda.time.DateTime;
import org.joda.time.format.DateTimeFormat;

import java.util.Arrays;
import java.util.List;
import java.util.UUID;

public class SQSExtendedClientExample {

// Create an Amazon S3 bucket with a random name.
private final static String S3_BUCKET_NAME = UUID.randomUUID() + "-"
        + DateTimeFormat.forPattern("yyMMdd-hhmmss").print(new DateTime());

public static void main(String[] args) {
```

```
/*
 * Create a new instance of the builder with all defaults (credentials
 * and region) set automatically. For more information, see
 * Creating Service Clients in the AWS SDK for Java Developer Guide.
 */
final AmazonS3 s3 = AmazonS3ClientBuilder.defaultClient();

/*
 * Set the Amazon S3 bucket name, and then set a lifecycle rule on the
 * bucket to permanently delete objects 14 days after each object's
 * creation date.
 */
final BucketLifecycleConfiguration.Rule expirationRule =
        new BucketLifecycleConfiguration.Rule();
expirationRule.withExpirationInDays(14).withStatus("Enabled");
final BucketLifecycleConfiguration lifecycleConfig =
        new BucketLifecycleConfiguration().withRules(expirationRule);

// Create the bucket and allow message objects to be stored in the bucket.
s3.createBucket(S3_BUCKET_NAME);
s3.setBucketLifecycleConfiguration(S3_BUCKET_NAME, lifecycleConfig);
System.out.println("Bucket created and configured.");

/*
 * Set the Amazon SQS extended client configuration with large payload
 * support enabled.
 */
final ExtendedClientConfiguration extendedClientConfig =
        new ExtendedClientConfiguration()
                .withLargePayloadSupportEnabled(s3, S3_BUCKET_NAME);

final AmazonSQS sqsExtended =
        new AmazonSQSExtendedClient(AmazonSQSClientBuilder
                .defaultClient(), extendedClientConfig);

/*
 * Create a long string of characters for the message object which will
 * be stored in the bucket.
 */
int stringLength = 300000;
char[] chars = new char[stringLength];
Arrays.fill(chars, 'x');
final String myLongString = new String(chars);
```

```
    // Create a message queue for this example.
    final String QueueName = "MyQueue" + UUID.randomUUID().toString();
    final CreateQueueRequest createQueueRequest =
            new CreateQueueRequest(QueueName);
    final String myQueueUrl = sqsExtended
            .createQueue(createQueueRequest).getQueueUrl();
    System.out.println("Queue created.");

    // Send the message.
    final SendMessageRequest myMessageRequest =
            new SendMessageRequest(myQueueUrl, myLongString);
    sqsExtended.sendMessage(myMessageRequest);
    System.out.println("Sent the message.");

    // Receive the message.
    final ReceiveMessageRequest receiveMessageRequest =
            new ReceiveMessageRequest(myQueueUrl);
    List<Message> messages = sqsExtended
            .receiveMessage(receiveMessageRequest).getMessages();

    // Print information about the message.
    for (Message message : messages) {
        System.out.println("\nMessage received.");
        System.out.println("  ID: " + message.getMessageId());
        System.out.println("  Receipt handle: " + message.getReceiptHandle());
        System.out.println("  Message body (first 5 characters): "
                + message.getBody().substring(0, 5));
    }

    // Delete the message, the queue, and the bucket.
    final String messageReceiptHandle = messages.get(0).getReceiptHandle();
    sqsExtended.deleteMessage(new DeleteMessageRequest(myQueueUrl,
            messageReceiptHandle));
    System.out.println("Deleted the message.");

    sqsExtended.deleteQueue(new DeleteQueueRequest(myQueueUrl));
    System.out.println("Deleted the queue.");

    deleteBucketAndAllContents(s3);
    System.out.println("Deleted the bucket.");
}

private static void deleteBucketAndAllContents(AmazonS3 client) {
```

```
    ObjectListing objectListing = client.listObjects(S3_BUCKET_NAME);

    while (true) {
        for (S3ObjectSummary objectSummary : objectListing
                .getObjectSummaries()) {
            client.deleteObject(S3_BUCKET_NAME, objectSummary.getKey());
        }

        if (objectListing.isTruncated()) {
            objectListing = client.listNextBatchOfObjects(objectListing);
        } else {
            break;
        }
    }

    final VersionListing list = client.listVersions(
            new ListVersionsRequest().withBucketName(S3_BUCKET_NAME));

    for (S3VersionSummary s : list.getVersionSummaries()) {
        client.deleteVersion(S3_BUCKET_NAME, s.getKey(), s.getVersionId());
    }

    client.deleteBucket(S3_BUCKET_NAME);
}
}
```

AWS Exemple de SDK pour Java 2.x : utilisation d'Amazon S3 pour gérer des messages Amazon SQS volumineux

L'exemple de AWS SDK pour Java 2.x suivant crée un compartiment Amazon S3 avec un nom aléatoire et ajoute une règle de cycle de vie pour supprimer définitivement les objets après 14 jours. Il crée également une file d'attente nommée `MyQueue` et envoie un message aléatoire qui est stocké dans un compartiment S3 et dont la taille est supérieure à 256 Ko pour la file d'attente. Enfin, le code récupère le message, renvoie des informations sur ce dernier et le supprime, ainsi que la file d'attente et le compartiment.

```
/*
 * Copyright 2010-2024 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
```

```
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

            import com.amazon.sqs.javamessaging.AmazonSQSExtendedClient;
import com.amazon.sqs.javamessaging.ExtendedClientConfiguration;
import org.joda.time.DateTime;
import org.joda.time.format.DateTimeFormat;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.BucketLifecycleConfiguration;
import software.amazon.awssdk.services.s3.model.CreateBucketRequest;
import software.amazon.awssdk.services.s3.model.DeleteBucketRequest;
import software.amazon.awssdk.services.s3.model.DeleteObjectRequest;
import software.amazon.awssdk.services.s3.model.ExpirationStatus;
import software.amazon.awssdk.services.s3.model.LifecycleExpiration;
import software.amazon.awssdk.services.s3.model.LifecycleRule;
import software.amazon.awssdk.services.s3.model.LifecycleRuleFilter;
import software.amazon.awssdk.services.s3.model.ListObjectVersionsRequest;
import software.amazon.awssdk.services.s3.model.ListObjectVersionsResponse;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Response;
import software.amazon.awssdk.services.s3.model.PutBucketLifecycleConfigurationRequest;
import software.amazon.awssdk.services.sqs.SqsClient;
import software.amazon.awssdk.services.sqs.model.CreateQueueRequest;
import software.amazon.awssdk.services.sqs.model.CreateQueueResponse;
import software.amazon.awssdk.services.sqs.model.DeleteMessageRequest;
import software.amazon.awssdk.services.sqs.model.DeleteQueueRequest;
import software.amazon.awssdk.services.sqs.model.Message;
import software.amazon.awssdk.services.sqs.model.ReceiveMessageRequest;
import software.amazon.awssdk.services.sqs.model.ReceiveMessageResponse;
import software.amazon.awssdk.services.sqs.model.SendMessageRequest;

import java.util.Arrays;
import java.util.List;
import java.util.UUID;
```

```
/**
 * Examples of using Amazon SQS Extended Client Library for Java 2.x
 *
 */
public class SqsExtendedClientExamples {
    // Create an Amazon S3 bucket with a random name.
    private final static String S3_BUCKET_NAME = UUID.randomUUID() + "-"
            + DateTimeFormat.forPattern("yyMMdd-hhmmss").print(new DateTime());

    public static void main(String[] args) {

        /*
         * Create a new instance of the builder with all defaults (credentials
         * and region) set automatically. For more information, see
         * Creating Service Clients in the AWS SDK for Java Developer Guide.
         */
        final S3Client s3 = S3Client.create();

        /*
         * Set the Amazon S3 bucket name, and then set a lifecycle rule on the
         * bucket to permanently delete objects 14 days after each object's
         * creation date.
         */
        final LifecycleRule lifeCycleRule = LifecycleRule.builder()
                .expiration(LifecycleExpiration.builder().days(14).build())
                .filter(LifecycleRuleFilter.builder().prefix("").build())
                .status(ExpirationStatus.ENABLED)
                .build();
        final BucketLifecycleConfiguration lifecycleConfig =
 BucketLifecycleConfiguration.builder()
                .rules(lifeCycleRule)
                .build();

        // Create the bucket and configure it
        s3.createBucket(CreateBucketRequest.builder().bucket(S3_BUCKET_NAME).build());

 s3.putBucketLifecycleConfiguration(PutBucketLifecycleConfigurationRequest.builder()
                .bucket(S3_BUCKET_NAME)
                .lifecycleConfiguration(lifecycleConfig)
                .build());
        System.out.println("Bucket created and configured.");

        // Set the Amazon SQS extended client configuration with large payload support
 enabled
```

```
        final ExtendedClientConfiguration extendedClientConfig = new
ExtendedClientConfiguration().withPayloadSupportEnabled(s3, S3_BUCKET_NAME);

        final SqsClient sqsExtended = new
AmazonSQSExtendedClient(SqsClient.builder().build(), extendedClientConfig);

        // Create a long string of characters for the message object
        int stringLength = 300000;
        char[] chars = new char[stringLength];
        Arrays.fill(chars, 'x');
        final String myLongString = new String(chars);

        // Create a message queue for this example
        final String queueName = "MyQueue-" + UUID.randomUUID();
        final CreateQueueResponse createQueueResponse =
sqsExtended.createQueue(CreateQueueRequest.builder().queueName(queueName).build());
        final String myQueueUrl = createQueueResponse.queueUrl();
        System.out.println("Queue created.");

        // Send the message
        final SendMessageRequest sendMessageRequest = SendMessageRequest.builder()
                .queueUrl(myQueueUrl)
                .messageBody(myLongString)
                .build();
        sqsExtended.sendMessage(sendMessageRequest);
        System.out.println("Sent the message.");

        // Receive the message
        final ReceiveMessageResponse receiveMessageResponse =
sqsExtended.receiveMessage(ReceiveMessageRequest.builder().queueUrl(myQueueUrl).build());
        List<Message> messages = receiveMessageResponse.messages();

        // Print information about the message
        for (Message message : messages) {
            System.out.println("\nMessage received.");
            System.out.println("  ID: " + message.messageId());
            System.out.println("  Receipt handle: " + message.receiptHandle());
            System.out.println("  Message body (first 5 characters): " +
message.body().substring(0, 5));
        }

        // Delete the message, the queue, and the bucket
        final String messageReceiptHandle = messages.get(0).receiptHandle();
```

```
 sqsExtended.deleteMessage(DeleteMessageRequest.builder().queueUrl(myQueueUrl).receiptHandle(me
        System.out.println("Deleted the message.");


 sqsExtended.deleteQueue(DeleteQueueRequest.builder().queueUrl(myQueueUrl).build());
        System.out.println("Deleted the queue.");

        deleteBucketAndAllContents(s3);
        System.out.println("Deleted the bucket.");

    }

    private static void deleteBucketAndAllContents(S3Client client) {
        ListObjectsV2Response listObjectsResponse =
 client.listObjectsV2(ListObjectsV2Request.builder().bucket(S3_BUCKET_NAME).build());

        listObjectsResponse.contents().forEach(object -> {

 client.deleteObject(DeleteObjectRequest.builder().bucket(S3_BUCKET_NAME).key(object.key()).bui
        });

        ListObjectVersionsResponse listVersionsResponse =
 client.listObjectVersions(ListObjectVersionsRequest.builder().bucket(S3_BUCKET_NAME).build());

        listVersionsResponse.versions().forEach(version -> {

 client.deleteObject(DeleteObjectRequest.builder().bucket(S3_BUCKET_NAME).key(version.key()).ve
        });


 client.deleteBucket(DeleteBucketRequest.builder().bucket(S3_BUCKET_NAME).build());
    }
}
```

Vous pouvez utiliser Apache Maven pour configurer et créer Amazon SQS Extended Client pour votre projet Java, ou pour créer le SDK lui-même. Spécifiez les modules individuels du SDK que vous utilisez dans votre application.

```
<properties>
    <aws-java-sdk.version>2.20.153</aws-java-sdk.version>
```

```
</properties>

<dependencies>
    <dependency>
      <groupId>software.amazon.awssdk</groupId>
      <artifactId>sqs</artifactId>
      <version>${aws-java-sdk.version}</version>
    </dependency>
    <dependency>
      <groupId>software.amazon.awssdk</groupId>
      <artifactId>s3</artifactId>
      <version>${aws-java-sdk.version}</version>
    </dependency>
    <dependency>
      <groupId>com.amazonaws</groupId>
      <artifactId>amazon-sqs-java-extended-client-lib</artifactId>
      <version>2.0.4</version>
    </dependency>

    <dependency>
      <groupId>joda-time</groupId>
      <artifactId>joda-time</artifactId>
      <version>2.12.6</version>
    </dependency>
</dependencies>
```

Gestion de messages Amazon SQS volumineux à l'aide de Python et Amazon S3

Vous pouvez utiliser la bibliothèque client étendue Amazon Simple Queue Service pour Python et Amazon Simple Storage Service pour gérer les messages Amazon SQS volumineux. Cela est particulièrement utile pour consommer des charges utiles de messages volumineuses, de 256 Ko à 2 Go. La bibliothèque enregistre la charge utile du message dans un compartiment Amazon S3 et envoie un message contenant une référence à l'objet Amazon S3 stocké à une file d'attente Amazon SQS.

Vous pouvez utiliser la bibliothèque client étendue pour Python pour effectuer les opérations suivantes :

- Spécifiez si les charges utiles sont toujours stockées dans Amazon S3 ou uniquement stockées dans S3 lorsque la taille de la charge utile dépasse 256 Ko
- Envoyer un message qui fait référence à un seul objet de message stocké dans un compartiment Amazon S3
- Récupérez l'objet de charge utile correspondant dans un compartiment Amazon S3
- Supprimer l'objet de charge utile correspondant d'un compartiment Amazon S3

Prérequis

Les conditions requises pour utiliser la bibliothèque client étendue Amazon SQS pour Python sont les suivantes :

- Un AWS compte avec les informations d'identification nécessaires. Pour créer un AWS compte, accédez à la page d'AWS accueil, puis choisissez Créer un AWS compte. Suivez les instructions à l'écran. Pour plus d'informations sur les informations d'identification, consultez la section Informations d'identification.
- Un AWS SDK : l'exemple de cette page utilise le SDK AWS Python Boto3. Pour installer et configurer le SDK, consultez la documentation du AWS SDK pour Python dans le guide du développeur du AWS SDK pour Python
- Python 3.x (ou version ultérieure) et. `pip`
- La bibliothèque client étendue Amazon SQS pour Python, disponible auprès de PyPI

> **Note**
>
> Vous pouvez utiliser la bibliothèque client étendue Amazon SQS pour Python pour gérer les messages Amazon SQS à l'aide d'Amazon S3 uniquement avec le AWS SDK pour Python. Vous ne pouvez pas le faire avec la AWS CLI, la console Amazon SQS, l'API HTTP Amazon SQS ou tout autre SDK. AWS

Configurer le stockage de messages

Le client Amazon SQS Extended utilise les attributs de message suivants pour configurer les options de stockage des messages Amazon S3 :

- `large_payload_support`: nom du compartiment Amazon S3 pour stocker les messages volumineux.
- `always_through_s3`: Si`True`, alors tous les messages sont stockés dans Amazon S3. Dans `False` le cas contraire, les messages inférieurs à 256 Ko ne seront pas sérialisés dans le compartiment s3. L'argument par défaut est `False`.
- `use_legacy_attribute`: Si tous `True` les messages publiés utilisent l'attribut de message réservé Legacy (`SQSLargePayloadSize`) au lieu de l'attribut de message réservé actuel (`ExtendedPayloadSize`).

Gestion de messages Amazon SQS volumineux avec la bibliothèque client étendue pour Python

L'exemple suivant crée un compartiment Amazon S3 avec un nom aléatoire. Il crée ensuite une file d'attente Amazon SQS nommée `MyQueue` et envoie un message qui est stocké dans un compartiment S3 et dont la taille est supérieure à 256 Ko à la file d'attente. Enfin, le code récupère le message, renvoie des informations sur ce dernier et le supprime, ainsi que la file d'attente et le compartiment.

```
import boto3
import sqs_extended_client

#Set the Amazon SQS extended client configuration with large payload.
sqs_extended_client = boto3.client("sqs", region_name="us-east-1")
sqs_extended_client.large_payload_support = "S3_BUCKET_NAME"
sqs_extended_client.use_legacy_attribute = False


# Create an SQS message queue for this example. Then, extract the queue URL.
queue = sqs_extended_client.create_queue(
    QueueName = "MyQueue"
)
queue_url = sqs_extended_client.get_queue_url(
    QueueName = "MyQueue"
)['QueueUrl']


# Create the S3 bucket and allow message objects to be stored in the bucket.
sqs_extended_client.s3_client.create_bucket(Bucket=sqs_extended_client.large_payload_support)
```

```
# Sending a large message
small_message = "s"
large_message = small_message * 300000 # Shall cross the limit of 256 KB

send_message_response = sqs_extended_client.send_message(
    QueueUrl=queue_url,
    MessageBody=large_message
)
assert send_message_response['ResponseMetadata']['HTTPStatusCode'] == 200

# Receiving the large message
receive_message_response = sqs_extended_client.receive_message(
    QueueUrl=queue_url,
    MessageAttributeNames=['All']
)
assert receive_message_response['Messages'][0]['Body'] == large_message
receipt_handle = receive_message_response['Messages'][0]['ReceiptHandle']

# Deleting the large message
# Set to True for deleting the payload from S3
sqs_extended_client.delete_payload_from_s3 = True
delete_message_response = sqs_extended_client.delete_message(
    QueueUrl=queue_url,
    ReceiptHandle=receipt_handle
)

assert delete_message_response['ResponseMetadata']['HTTPStatusCode'] == 200

# Deleting the queue
delete_queue_response = sqs_extended_client.delete_queue(
    QueueUrl=queue_url
)

assert delete_queue_response['ResponseMetadata']['HTTPStatusCode'] == 200
```

Configuration des files d'attente Amazon SQS à l'aide de la console Amazon SQS

Utilisez la console Amazon SQS pour configurer et gérer les files d'attente et les fonctionnalités Amazon Simple Queue Service (Amazon SQS). Vous pouvez également utiliser la console pour configurer des fonctionnalités telles que le chiffrement côté serveur, associer une file d'attente de lettres mortes à votre file d'attente ou définir un déclencheur pour appeler une fonction. AWS Lambda

Rubriques

Contrôle d'accès basé sur les attributs pour Amazon SQS

Qu'est-ce que le contrôle d'accès basé sur les attributs (ABAC) ?

Le contrôle d'accès basé sur les attributs (ABAC) est un processus d'autorisation qui définit les autorisations en fonction des balises associées aux utilisateurs et aux ressources. AWS L'ABAC fournit un contrôle d'accès granulaire et flexible basé sur des attributs et des valeurs, réduit les risques de sécurité liés aux stratégies reconfigurées basées sur les rôles et centralise l'audit et la

gestion des stratégies d'accès. Pour plus de détails sur l'ABAC, consultez Qu'est-ce que l'ABAC pour AWS ? dans le Guide de l'utilisateur IAM.

Amazon SQS prend en charge l'ABAC en vous permettant de contrôler l'accès à vos files d'attente Amazon SQS en fonction des balises et des alias associés à une file d'attente Amazon SQS. Les clés de condition de balise et d'alias qui activent l'ABAC dans Amazon SQS autorisent les mandataires IAM à utiliser les files d'attente Amazon SQS sans modifier les stratégies ni gérer les octrois.

Avec ABAC, vous pouvez utiliser des balises pour configurer les autorisations et les stratégies d'accès IAM pour vos files d'attente Amazon SQS, ce qui vous permet de mettre à l'échelle votre gestion des autorisations. Vous pouvez créer une stratégie d'autorisation unique dans IAM à l'aide de balises que vous ajoutez à chaque rôle commercial, sans avoir à mettre à jour la stratégie chaque fois que vous ajoutez une nouvelle ressource. Vous pouvez également associer des balises aux mandataires IAM pour créer une stratégie ABAC. Vous pouvez concevoir des stratégies ABAC pour autoriser les opérations Amazon SQS lorsque la balise du rôle d'utilisateur IAM qui effectue l'appel correspond à la balise de file d'attente Amazon SQS. Pour en savoir plus sur le balisage AWS, consultez les sections Stratégies de AWS balisage et. Balises de répartition des coûts Amazon SQS

> **Note**
>
> ABAC pour Amazon SQS est actuellement disponible dans AWS toutes les régions commerciales où Amazon SQS est disponible, avec les exceptions suivantes :
>
> - Asie-Pacifique (Hyderabad)
> - Asie-Pacifique (Melbourne)
> - Europe (Espagne)
> - Europe (Zurich)

Pourquoi utiliser l'ABAC dans Amazon SQS ?

Voici quelques avantages liés à l'utilisation de l'ABAC dans Amazon SQS :

- L'ABAC pour Amazon SQS nécessite moins de stratégies d'autorisation. Vous n'avez pas besoin de créer une stratégie pour chaque activité professionnelle. Vous pouvez utiliser des balises de ressource et de demande qui s'appliquent à plusieurs files d'attente, ce qui réduit la charge opérationnelle.

- Utilisez l'ABAC pour mettre rapidement à l'échelle les équipes. Les autorisations d'accès aux nouvelles ressources sont automatiquement accordées en fonction des balises lorsque les ressources sont correctement balisées lors de leur création.
- Utilisez les autorisations sur le mandataire IAM pour restreindre l'accès aux ressources. Vous pouvez créer des balises pour le mandataire IAM et les utiliser pour restreindre l'accès à des actions spécifiques correspondant aux balises du mandataire IAM. Cela vous permet d'automatiser le processus d'octroi des autorisations de demande.
- Suivez les personnes qui accèdent à vos ressources. Vous pouvez déterminer l'identité d'une session en consultant les attributs utilisateur dans AWS CloudTrail.

Rubriques

Clés de condition pour Amazon SQS

Vous pouvez utilisez les clés de condition suivantes pour contrôler les actions de fonction :

Clé de condition ABAC	Description	Type de stratégie	Opérations Amazon SQS
lois : ResourceTag	La balise (clé et valeur) de la file d'attente Amazon SQS correspond à la balise (clé et valeur) ou au modèle de balise dans la stratégie.	Politique IAM uniquement	Opérations sur les ressources de file d'attente Amazon SQS
lois : RequestTag	La balise (clé et valeur) dans les opération	Stratégie de file d'attente et stratégies IAM	TagQueue, UntagQueue, CreateQueue

Clé de condition ABAC	Description	Type de stratégie	Opérations Amazon SQS
	s de ressource de file d'attente Amazon SQS correspond à la balise (clé et valeur) ou au modèle de balise dans la stratégie.		
lois : TagKeys	Dans la demande, les clés de balise correspondent à celles de la politique.	Stratégie de file d'attente et stratégies IAM	TagQueue, UntagQueue, CreateQueue

Balisage pour le contrôle d'accès dans Amazon SQS

Voici un exemple d'utilisation des balises pour le contrôle d'accès. La stratégie IAM limite un utilisateur IAM à toutes les actions Amazon SQS pour toutes les files d'attente qui incluent une balise de ressource indiquant l'environnement clé et la production de valeur. Pour plus d'informations, voir Contrôle d'accès basé sur les attributs avec balises et AWS Organizations.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAccessForProd",
      "Effect": "Deny",
      "Action": "sqs:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/environment": "prod"
        }
      }
    }
  ]
}
```

Création d'utilisateurs IAM et de files d'attente Amazon SQS

Les exemples suivants expliquent comment créer une politique ABAC pour contrôler l'accès à Amazon SQS à l'aide AWS Management Console du et. AWS CloudFormation

À l'aide du AWS Management Console

Créer un utilisateur IAM

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/iam/.
2. Choisissez Utilisateur dans le panneau de navigation de gauche.
3. Choisissez Ajouter des utilisateurs et saisissez un nom dans la zone de texte Nom d'utilisateur.
4. Sélectionnez Clé d'accès - Accès programmatique, puis choisissez Suivant : Autorisations.
5. Choisissez Suivant : balises.
6. Ajouter la clé d'étiquette en tant que `environment` et la valeur de balise en tant que `beta`.
7. Choisissez Suivant : Autorisations, puis Créer un utilisateur.
8. Conservez votre ID de clé d'accès et votre clé d'accès secrète dans un emplacement sécurisé.

Ajouter des autorisations d'utilisateur IAM

1. Sélectionnez l'utilisateur IAM que vous avez créé.
2. Sélectionnez Ajouter une politique en ligne.
3. Dans l'onglet JSON, collez la stratégie suivante :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessForSameResTag",
      "Effect": "Allow",
      "Action": [
        "sqs:SendMessage",
        "sqs:ReceiveMessage",
        "sqs:DeleteMessage"
      ],
      "Resource": "*",
      "Condition": {
```

```
        "StringEquals": {
          "aws:ResourceTag/environment": "${aws:PrincipalTag/environment}"
        }
      }
    },
    {
      "Sid": "AllowAccessForSameReqTag",
      "Effect": "Allow",
      "Action": [
        "sqs:CreateQueue",
        "sqs:DeleteQueue",
        "sqs:SetQueueAttributes",
        "sqs:tagqueue"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/environment": "${aws:PrincipalTag/environment}"
        }
      }
    },
    {
      "Sid": "DenyAccessForProd",
      "Effect": "Deny",
      "Action": "sqs:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/stage": "prod"
        }
      }
    }
  ]
}
```

4. Choisissez Examiner une politique.
5. Choisissez Créer une politique.

En utilisant AWS CloudFormation

Utilisez l'exemple de AWS CloudFormation modèle suivant pour créer un utilisateur IAM associé à une politique intégrée et à une file d'attente Amazon SQS :

```
AWSTemplateFormatVersion: "2010-09-09"
Description: "CloudFormation template to create IAM user with custom inline policy"
Resources:
    IAMPolicy:
        Type: "AWS::IAM::Policy"
        Properties:
            PolicyDocument: |
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Sid": "AllowAccessForSameResTag",
                            "Effect": "Allow",
                            "Action": [
                                "sqs:SendMessage",
                                "sqs:ReceiveMessage",
                                "sqs:DeleteMessage"
                            ],
                            "Resource": "*",
                            "Condition": {
                                "StringEquals": {
                                    "aws:ResourceTag/environment": "${aws:PrincipalTag/
environment}"
                                }
                            }
                        },
                        {
                            "Sid": "AllowAccessForSameReqTag",
                            "Effect": "Allow",
                            "Action": [
                                "sqs:CreateQueue",
                                "sqs:DeleteQueue",
                                "sqs:SetQueueAttributes",
                                "sqs:tagqueue"
                            ],
                            "Resource": "*",
                            "Condition": {
                                "StringEquals": {
                                    "aws:RequestTag/environment": "${aws:PrincipalTag/
environment}"
                                }
                            }
                        },
```

```
                    {
                        "Sid": "DenyAccessForProd",
                        "Effect": "Deny",
                        "Action": "sqs:*",
                        "Resource": "*",
                        "Condition": {
                            "StringEquals": {
                                "aws:ResourceTag/stage": "prod"
                            }
                        }
                    }
                ]
            }

        Users:
          - "testUser"
        PolicyName: tagQueuePolicy

IAMUser:
    Type: "AWS::IAM::User"
    Properties:
        Path: "/"
        UserName: "testUser"
        Tags:
          -
            Key: "environment"
            Value: "beta"
```

Test du contrôle d'accès basé sur les attributs

Les exemples suivants vous montrent comment tester le contrôle d'accès basé sur les attributs dans Amazon SQS.

Créer une file d'attente avec la clé de balise définie sur environment et la valeur de balise définie sur prod

Exécutez cette commande AWS CLI pour tester la création de la file d'attente avec la clé de balise définie sur environnement et la valeur de balise définie sur prod. Si vous n'avez pas de AWS CLI, vous pouvez la télécharger et la configurer pour votre machine.

```
aws sqs create-queue --queue-name prodQueue –region us-east-1 –tags "environment=prod"
```

Vous recevez une erreur `AccessDenied` de la part du point de terminaison Amazon SQS :

```
An error occurred (AccessDenied) when calling the CreateQueue operation: Access to the
 resource <queueUrl> is denied.
```

Cela est dû au fait que la valeur de balise de l'utilisateur IAM ne correspond pas à la balise transmise lors de l'appel d'API `CreateQueue`. N'oubliez pas que nous avons appliqué une balise à l'utilisateur IAM avec la clé définie sur `environment` et la valeur définie sur `beta`.

Créer une file d'attente avec la clé de balise définie sur environment et la valeur de balise définie sur beta

Exécutez cette commande CLI pour tester la création d'une file d'attente avec la clé de balise définie sur `environment` et la valeur de balise définie sur `beta`.

```
aws sqs create-queue --queue-name betaQueue –region us-east-1 –tags "environment=beta"
```

Vous recevez un message confirmant la création réussie de la file d'attente, similaire à celui ci-dessous.

```
{
"QueueUrl": "<queueUrl>“
}
```

Envoi d'un message à une file d'attente

Exécutez cette commande CLI pour tester l'envoi d'un message à une file d'attente.

```
aws sqs send-message --queue-url <queueUrl> --message-body testMessage
```

La réponse indique que le message a été correctement envoyé à la file d'attente Amazon SQS. L'autorisation d'utilisateur IAM vous permet d'envoyer un message à une file d'attente comportant une balise `beta`. La réponse inclut `MD5OfMessageBody` et `MessageId` contenant le message.

```
{
"MD5OfMessageBody": "<MD5OfMessageBody>",
"MessageId": "<MessageId>"
}
```

Configuration des paramètres de file d'attente à l'aide de la console Amazon SQS

Lorsque vous créez ou modifiez une file d'attente, vous pouvez configurer les paramètres suivants :

- Délai de visibilité : durée pendant laquelle un message reçu d'une file d'attente (par un consommateur) ne sera pas visible pour les autres consommateurs de messages. Pour plus d'informations, consultez Délai de visibilité.

  > **Note**
  >
  > L'utilisation de la console pour configurer le délai de visibilité permet de configurer la valeur du délai pour tous les messages de la file d'attente. Pour configurer le délai d'expiration pour un ou plusieurs messages, vous devez utiliser l'un des AWS SDK.

- Période de conservation des messages : durée pendant laquelle Amazon SQS conserve les messages qui restent dans la file d'attente. Par défaut, la file d'attente conserve les messages pendant quatre jours. Vous pouvez configurer une file d'attente pour conserver les messages jusqu'à 14 jours. Pour plus d'informations, consultez Période de conservation des messages.
- Retard de livraison : durée pendant laquelle Amazon SQS retardera la livraison d'un message ajouté à la file d'attente. Pour plus d'informations, consultez Retard de livraison.
- Taille maximale des messages : taille maximale des messages pour cette file d'attente. Pour plus d'informations, consultez Taille maximale des messages.
- Temps d'attente des messages de réception : durée maximale pendant laquelle Amazon SQS attend que les messages soient disponibles une fois que la file d'attente a reçu une demande de réception. Pour plus d’informations, consultez Recherches courtes et longues sur Amazon SQS.
- Activer la déduplication basée sur le contenu : Amazon SQS peut créer automatiquement des identifiants de déduplication en fonction du corps du message. Pour plus d’informations, consultez Commencer à utiliser les files d'attente FIFO dans Amazon SQS.
- Activer le FIFO à haut débit : à utiliser pour activer le débit élevé pour les messages de la file d'attente. Le choix de cette option modifie les options associées (Portée de la déduplication et Limite de débit FIFO) en fonction des paramètres requis pour activer un débit élevé pour les files d'attente FIFO. Pour plus d'informations, consultez Débit élevé pour les files d'attente FIFO dans Amazon SQS et Quotas de messages Amazon SQS.

- Stratégie d'autorisation de redirection : définit les files d'attente source pouvant utiliser cette file d'attente comme file d'attente de lettres mortes. Pour plus d'informations, consultez Utilisation de files d'attente contenant des lettres mortes dans Amazon SQS .

Pour configurer des paramètres de file d'attente pour une file d'attente existante (console)

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
2. Dans le volet de navigation, choisissez Files d'attente. Choisissez une file d'attente, puis sélectionnez Modifier.
3. Accédez à la section Configuration.
4. Pour le Délai de visibilité, saisissez la durée et les unités. La plage est comprise entre 0 seconde et 12 heures. La valeur par défaut est de 30 secondes.
5. Pour Période de conservation des messages, saisissez la durée et les unités. La plage est comprise entre 1 minute et 14 jours. La valeur par défaut est de 4 jours.
6. Pour une file d'attente standard, saisissez une valeur pour Temps d'attente du message de réception. La plage est comprise entre 0 et 20 secondes. La valeur par défaut est 0 seconde, qui permet de définir la recherche courte. Toute valeur différente de zéro définit une recherche longue.
7. Pour Retard de diffusion, saisissez la durée et les unités. La plage est comprise entre 0 seconde et 15 minutes. La valeur par défaut est de 0 seconde.
8. Pour Taille maximale du message, saisissez une valeur. La plage est comprise entre 1 et 256 Ko. La valeur par défaut est de 256 Ko.
9. Pour une file d'attente FIFO, choisissez Activer la déduplication basée sur le contenu pour activer cette option. Par défaut, ce paramètre est désactivé.
10. (Facultatif) Pour qu'une file d'attente FIFO permette un débit plus élevé pour l'envoi et la réception de messages dans la file d'attente, choisissez Activer le FIFO à haut débit.

    Le choix de cette option modifie les options associées (Portée de la déduplication et Limite de débit FIFO) en fonction des paramètres requis pour activer un débit élevé pour les files d'attente FIFO. Si vous modifiez l'un des paramètres requis pour utiliser le FIFO à débit élevé, le débit normal est effectif pour la file d'attente et la déduplication se produit comme indiqué. Pour plus d'informations, consultez Débit élevé pour les files d'attente FIFO dans Amazon SQS et Quotas de messages Amazon SQS.

11. Pour la Stratégie d'autorisation de redirection, choisissez Activé. Sélectionnez l'une des options suivantes : Tout autoriser (par défaut), Par file d'attente ou Refuser tout. Lorsque vous choisissez Par file d'attente, spécifiez une liste de 10 files d'attente source maximum en fonction de l'Amazon Resource Name (ARN).
12. Lorsque vous avez fini de configurer les paramètres de la file d'attente, choisissez Enregistrer.

Configuration de la stratégie d'accès

Lorsque vous modifiez une file d'attente, vous pouvez configurer sa stratégie d'accès.

La stratégie d'accès définit les comptes, les utilisateurs et les rôles qui peuvent accéder à la file d'attente. La stratégie d'accès définit également les actions (telles que `SendMessage`, `ReceiveMessage` ou`DeleteMessage`) auxquelles les utilisateurs peuvent accéder. La stratégie par défaut permet uniquement au propriétaire de la file d'attente d'envoyer et de recevoir des messages.

Pour configurer la stratégie d'accès pour une file d'attente existante (console)

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
2. Dans le volet de navigation, choisissez Files d'attente.
3. Choisissez une file d'attente, puis sélectionnez Modifier.
4. Accédez à la section stratégie d'accès.
5. Modifiez les instructions de stratégie d'accès dans la zone de saisie. Pour en savoir plus sur les instructions de stratégie d'accès, consultez Gestion des identités et des accès dans Amazon SQS.
6. Lorsque vous avez terminé de configurer la stratégie d'accès, choisissez Enregistrer.

Configuration du chiffrement côté serveur pour une file d'attente à l'aide de clés de chiffrement gérées par SQL

Outre l'option de chiffrement côté serveur (SSE) géré par Amazon SQS par défaut, le SSE géré par Amazon SQS (SSE-SQS) vous permet de créer un chiffrement géré personnalisé côté serveur qui utilise des clés de chiffrement gérées par SQS pour protéger les données sensibles envoyées via des files d'attente de messages. Avec le SSE-SQS, vous n'avez pas besoin de créer et de gérer des clés de chiffrement, ni de modifier votre code pour chiffrer vos données. Le SSE-SQS vous permet

de transmettre des données en toute sécurité et de respecter les exigences réglementaires et de conformité strictes en matière de chiffrement, sans frais supplémentaires.

Le SSE-SQS protège les données au repos à l'aide d'un chiffrement Advanced Encryption Standard 256 bits (AES-256). SSE chiffre les messages une fois reçus par Amazon SQS. Amazon SQS stocke les messages sous forme chiffrée et ne les déchiffre que lorsqu'ils sont envoyés à un consommateur autorisé.

> **Note**
>
> - L'option SSE par défaut n'est efficace que lorsque vous créez une file d'attente sans spécifier d'attributs de chiffrement.
> - Amazon SQS vous permet de désactiver le chiffrement de toutes les files d'attente. Par conséquent, la désactivation du KMS-SSE n'activera pas automatiquement le SQS-SSE. Si vous souhaitez activer le SQS-SSE après avoir désactivé le KMS-SSE, vous devez ajouter un changement d'attribut dans la demande.

Pour configurer le chiffrement SSE-SQS pour une file d'attente (console)

> **Note**
>
> Toute nouvelle file d'attente créée à l'aide du point de terminaison HTTP (non-TLS) n'activera pas le chiffrement SSE-SQS par défaut. La création de files d'attente Amazon SQS à l'aide de points de terminaison HTTPS ou Signature Version 4 constitue une bonne pratique en matière de sécurité.

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
2. Dans le volet de navigation, choisissez Files d'attente.
3. Choisissez une file d'attente, puis sélectionnez Modifier.
4. Développez Chiffrement.
5. Sous Chiffrement côté serveur, choisissez Activer (par défaut).

> **Note**
>
> Lorsque SSE est activé, les demandes anonymes `SendMessage` et `ReceiveMessage` adressées à la file d'attente chiffrée sont rejetées. Les bonnes pratiques de sécurité d'Amazon SQS recommandent de ne pas utiliser de demandes anonymes. Si vous souhaitez envoyer des demandes anonymes à une file d'attente Amazon SQS, veillez à désactiver SSE.

6. Sélectionnez la clé Amazon SQS (SSE-SQS). Aucuns frais supplémentaires ne vous sont facturés pour l'utilisation de cette option.
7. Choisissez Enregistrer.

Configuration du chiffrement côté serveur pour une file d'attente à l'aide de la console Amazon SQS

Pour protéger les données contenues dans les messages d'une file d'attente, Amazon SQS a activé le chiffrement côté serveur (SSE) par défaut pour toutes les files d'attente nouvellement créées. Amazon SQS s'intègre au service de gestion des clés Amazon Web Services (Amazon Web Services KMS) afin de gérer les clés KMS pour le chiffrement côté serveur (SSE). Pour plus d'informations sur l'utilisation du chiffrement SSE, consultez Chiffrement au repos dans Amazon SQS.

La clé KMS que vous attribuez à votre file d'attente doit avoir une stratégie de clé qui inclut des autorisations pour tous les mandataires autorisés à utiliser la file d'attente. Pour plus d'informations, consultez Gestion des clés.

Si vous n'êtes pas le propriétaire de la clé KMS, ou si vous vous connectez avec un compte n'ayant pas les autorisations `kms:ListAliases` et `kms:DescribeKey`, vous ne pouvez pas afficher les informations relatives à la clé KMS sur la console Amazon SQS. Demandez au propriétaire de la clé KMS de vous accorder ces autorisations. Pour plus d'informations, consultez Gestion des clés.

Lorsque vous créez ou modifiez une file d'attente, vous pouvez configurer le SSE-KMS.

Pour configurer le SSE-KMS pour une file d'attente existante (console)

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
2. Dans le volet de navigation, choisissez Files d'attente.

3. Choisissez une file d'attente, puis sélectionnez Modifier.
4. Développez Chiffrement.
5. Sous Chiffrement côté serveur, choisissez Activer (par défaut).

> **Note**
>
> Lorsque SSE est activé, les demandes anonymes `SendMessage` et `ReceiveMessage` adressées à la file d'attente chiffrée sont rejetées. Les bonnes pratiques de sécurité d'Amazon SQS recommandent de ne pas utiliser de demandes anonymes. Si vous souhaitez envoyer des demandes anonymes à une file d'attente Amazon SQS, veillez à désactiver SSE.

6. Sélectionnez CléAWS Key Management Service (SSE-KMS).

   La console affiche la Description, le Compte et l'ARN de la clé KMS.
7. Spécifiez l'ID de clé KMS pour la file d'attente. Pour plus d’informations, consultez Termes clés.
   a. Choisissez l'option Choisir un alias de clé KMS.
   b. La clé par défaut est la clé KMS gérée par Amazon Web Services pour Amazon SQS. Pour utiliser cette clé, choisissez-la dans la liste des clés KMS.
   c. Pour utiliser une clé KMS personnalisée depuis votre compte Amazon Web Services, choisissez-la dans la liste des clés KMS. Pour obtenir des instructions sur la création de clés KMS personnalisées, consultez Création de clés dans le Guide du développeur Amazon Web Services Key Management Service.
   d. Pour utiliser une clé KMS personnalisée qui ne figure pas dans la liste, ou une clé KMS personnalisée provenant d'un autre compte Amazon Web Services, choisissez Saisir l'alias de la clé KMS, puis saisissez l'Amazon Resource Name (ARN) de la clé KMS.
8. (Facultatif) Pour la Période de réutilisation des clés de données, spécifiez une valeur comprise entre 1 minute et 24 heures. La valeur par défaut est de 5 minutes. Pour plus d’informations, consultez Présentation de la période de réutilisation des clés de données.
9. Lorsque vous avez terminé de configurer le SSE-KMS, choisissez Enregistrer.

Configuration des balises de répartition des coûts pour une file d'attente à l'aide de la console Amazon SQS

Pour organiser et identifier vos files d'attente Amazon SQS, vous pouvez leur ajouter des balises de répartition des coûts. Pour plus d'informations, consultez Balises de répartition des coûts Amazon SQS.

Sur la page Détails d'une file d'attente, l'onglet Balisage affiche les balises de la file d'attente.

Lorsque vous créez ou modifiez une file d'attente, vous pouvez configurer des balises à lui ajouter.

Pour configurer les balises d'une file d'attente existante (console)

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
2. Dans le volet de navigation, choisissez Files d'attente.
3. Choisissez une file d'attente, puis sélectionnez Modifier.
4. Faites défiler jusqu'à la section Balises.
5. Ajouter, modifier ou supprimer des balises de file d'attente :
    a. Pour ajouter une balise, choisissez Ajouter une balise, saisissez une clé et une valeur, puis choisissez Ajouter une nouvelle balise.
    b. Pour mettre à jour une balise, modifiez sa clé et sa valeur.
    c. Pour supprimer une balise, choisissez Supprimer en regard de sa paire clé-valeur.
6. Lorsque vous avez terminé de configurer les balises, choisissez Enregistrer.

Abonnement d'une file d'attente à une rubrique Amazon SNS à l'aide de la console Amazon SQS

Vous pouvez abonner une ou plusieurs files d'attente Amazon SQS à une rubrique Amazon Simple Notification Service (Amazon SNS). Lorsque vous publiez un message dans une rubrique, Amazon SNS envoie le message à chaque file d'attente qui y est abonnée. Amazon SQS gère l'abonnement et les autorisations nécessaires. Pour plus d'informations sur Amazon SNS, consultez Qu'est-ce qu'Amazon SNS ? dans le Guide du développeur Amazon Simple Notification Service.

Lorsque vous abonnez une file d'attente Amazon SQS à une rubrique Amazon SNS, Amazon SNS utilise HTTPS pour transférer les messages à Amazon SQS. Pour plus d'informations sur l'utilisation

d'Amazon SNS avec des files d'attente Amazon SQS chiffrées, consultez Configuration des autorisations KMS pour les AWS services.

> **Important**
>
> Amazon SQS prend en charge un maximum de 20 instructions par stratégie d'accès. L'abonnement à une rubrique Amazon SNS ajoute ce type d'instruction. Le dépassement de ce montant entraînera l'échec de la livraison de l'abonnement à la rubrique.

Pour abonner une file d'attente à une rubrique SNS (console)

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
2. Dans le volet de navigation, choisissez Files d'attente.
3. Dans la liste des files d'attente, choisissez la file d'attente à abonner à la rubrique SNS.
4. Dans Actions, choisissez Subscribe to Amazon SNS topic (Abonner à la rubrique Amazon SNS).
5. Dans le menu Spécifiez une rubrique Amazon SNS disponible pour cette file d'attente, choisissez la rubrique SNS pour votre file d'attente.

   Si la rubrique SNS n'est pas répertoriée dans le menu, choisissez Saisir l'ARN de la rubrique Amazon SNS, puis saisissez l'Amazon Resource Name (ARN) de la rubrique.
6. Choisissez Enregistrer.
7. Pour vérifier le résultat de l'abonnement, publiez quelque chose dans la rubrique et consulter le message que celle-ci envoie à la file d'attente. Pour plus d'informations, consultez Diffusion de messages Amazon SNS dans le Guide du développeur Amazon Simple Notification Service.

Si votre file d'attente Amazon SQS et votre rubrique SNS sont différentes Comptes AWS, le propriétaire de la rubrique doit d'abord confirmer l'abonnement. Pour plus d'informations, consultez Confirmer l'abonnement dans le Guide du développeur Amazon Simple Notification Service.

Pour plus d'informations sur l'abonnement à une rubrique SNS interrégionale, consultez la section Envoyer des messages Amazon SNS à une file d'attente ou à une AWS Lambda fonction Amazon SQS dans une autre région dans le manuel Amazon Simple Notification Service Developer Guide

Configuration d'une file d'attente Amazon SQS pour déclencher une fonction AWS Lambda

Vous pouvez utiliser une AWS Lambda fonction pour traiter les messages dans une file d'attente Amazon SQS. Lambda interroge la file d'attente et invoque votre fonction Lambda de manière synchrone avec un événement contenant les messages de la file d'attente. Pour autoriser votre fonction à traiter chaque lot d'enregistrements, définissez le délai de visibilité de la file d'attente source à au moins six fois le délai d'attente que vous configurez sur votre fonction. Le délai supplémentaire permet à Lambda d'effectuer une nouvelle tentative si l'exécution de la fonction est limitée pendant le traitement d'un lot précédent.

Vous pouvez spécifier une autre file d'attente qui servira de file d'attente de lettres mortes pour les messages que votre fonction Lambda ne peut pas traiter.

Une fonction Lambda peut traiter des éléments de plusieurs files d'attente (avec une source d'événement Lambda pour chaque file d'attente). Vous pouvez utiliser la même file d'attente avec plusieurs fonctions Lambda.

Si vous associez une file d'attente chiffrée à une fonction Lambda mais que Lambda n'interroge pas les messages, ajoutez l'autorisation `kms:Decrypt` à votre rôle d'exécution Lambda.

Notez les restrictions suivants :

- Votre file d'attente et la fonction Lambda doivent se trouver dans la même AWS région.
- Une file d'attente chiffrée qui utilise la clé par défaut (clé KMS AWS gérée pour Amazon SQS) ne peut pas appeler de fonction Lambda dans un autre. Compte AWS

Pour plus d'informations sur l'implémentation de la fonction Lambda, consultez la section Utilisation AWS Lambda avec Amazon SQS dans AWS Lambda le manuel du développeur.

Prérequis

Pour configurer les déclencheurs de la fonction Lambda, vous devez respecter les conditions requises suivantes :

- Si vous faites appel à un utilisateur, votre rôle Amazon SQS doit inclure les autorisations suivantes :
  - `lambda:CreateEventSourceMapping`

  - `lambda:ListEventSourceMappings`
  - `lambda:ListFunctions`
- Le rôle d'exécution Lambda doit inclure les autorisations suivantes :
  - `sqs:DeleteMessage`
  - `sqs:GetQueueAttributes`
  - `sqs:ReceiveMessage`
- Si vous associez une file d'attente chiffrée à une fonction Lambda, ajoutez l'autorisation `kms:Decrypt` à votre rôle d'exécution Lambda.

Pour plus d’informations, consultez Présentation de la gestion de l'accès dans Amazon SQS.

Pour configurer une file d'attente afin de déclencher une fonction Lambda (console)

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
2. Dans le volet de navigation, choisissez Files d'attente.
3. Sur la page Files d'attente, choisissez la file d'attente à configurer.
4. Sur la page de la file d'attente, choisissez l'onglet Déclencheurs de fonction Lambda.
5. Sur la page Déclencheurs de fonction Lambda, choisissez un déclencheur de fonction Lambda.

   Si la liste n'inclut pas le déclencheur de fonction Lambda dont vous avez besoin, choisissez Configurer le déclencheur de fonction Lambda. Saisissez l'Amazon Resource Name (ARN) de la fonction Lambda ou choisissez une ressource existante. Ensuite, choisissez Save (Enregistrer).
6. Choisissez Enregistrer. La console enregistre la configuration et affiche la page Détails de la file d'attente.

   Sur la page Détails, l'onglet Déclencheurs de fonction Lambda affiche la fonction Lambda et son statut. L'association de la fonction Lambda à votre file d'attente peut prendre environ 1 minute.
7. Pour vérifier le résultat de la configuration, envoyez un message à votre file d'attente et affichez la fonction Lambda déclenchée dans la console Lambda.

Automatiser les notifications envoyées par les AWS services à Amazon SQS à l'aide d'Amazon EventBridge

Amazon vous EventBridge permet d'automatiser les AWS services et de répondre aux événements du système tels que les problèmes de disponibilité des applications ou les modifications des ressources. Les événements AWS liés aux services sont diffusés EventBridge presque en temps réel. Vous pouvez écrire des règles simples pour préciser les événements qui vous intéressent et les actions automatisées à effectuer quand un événement correspond à une règle.

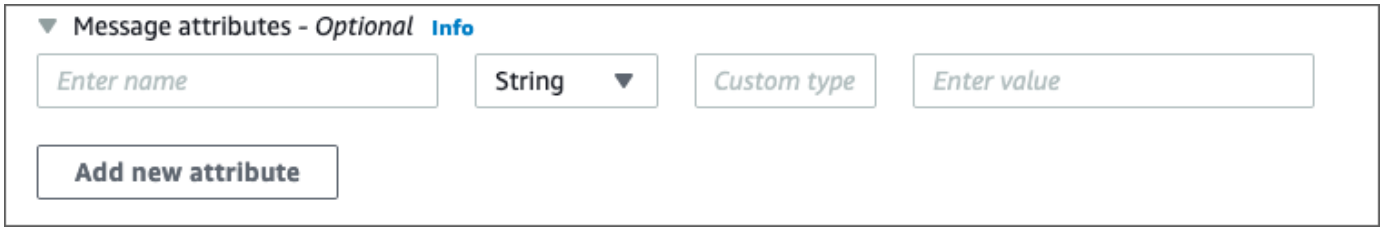
EventBridge vous permet de définir diverses cibles, telles que les files d'attente Amazon SQS standard et FIFO, qui reçoivent des événements au format JSON. Pour plus d'informations, consultez les EventBridgecibles Amazon dans le guide de EventBridge l'utilisateur Amazon.

Envoi d'un message avec des attributs

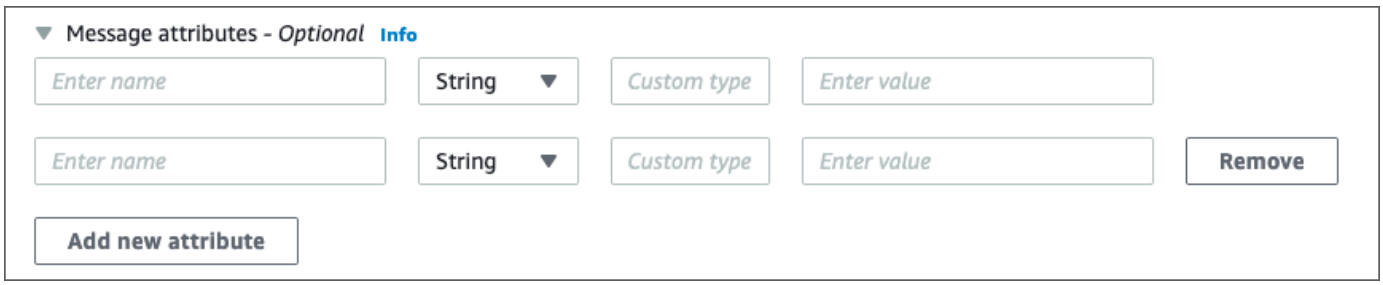
Pour les files d'attente standard et FIFO, vous pouvez inclure des métadonnées structurées (telles que des horodatages, des données géospatiales, des signatures et des identifiants) dans les messages. Pour plus d'informations, consultez Attributs de message Amazon SQS.

Pour envoyer un message contenant des attributs à une file d'attente à l'aide de la console Amazon SQS

1. Ouvrez la console Amazon SQS à l'adresse https://console.aws.amazon.com/sqs/.
2. Dans le volet de navigation, choisissez Files d'attente.
3. Sur la page Files d'attente, choisissez une file d'attente.
4. Choisissez Envoyer et recevoir des messages.
5. Saisissez les paramètres d'attribut du message.
    a. Dans la zone de texte Nom, saisissez un nom unique comportant jusqu'à 256 caractères.
    b. Pour le type d'attribut, choisissez Chaîne, Nombre ou Binaire.
    c. (Facultatif) Saisissez un type de données personnalisé. Par exemple, vous pouvez ajouter **`byte`**, **`int`** ou **`float`** en tant que types de données personnalisés pour Nombre.
    d. Dans la zone de texte de la valeur, saisissez la valeur de l'attribut de message.



6. Pour ajouter un autre attribut de message, sélectionnez Ajouter un nouvel attribut.



7. Vous pouvez modifier les valeurs d'attribut à tout moment avant d'envoyer le message.
8. Pour supprimer un attribut, choisissez Supprimer. Pour supprimer le premier attribut, fermez les attributs du message.
9. Lorsque vous avez fini d'ajouter des attributs au message, choisissez Envoyer un message. Votre message est envoyé et la console affiche un message de réussite. Pour afficher les informations relatives aux attributs du message envoyé, choisissez Afficher les détails. Choisissez Terminé pour fermer la boîte de dialogue Détails du message.

Bonnes pratiques relatives à Amazon SQS

Ces bonnes pratiques peuvent vous aider à tirer le meilleur d'Amazon SQS.

Rubriques

Recommandations pour les files d'attente Amazon SQS standard et FIFO

Les bonnes pratiques suivantes peuvent vous aider à réduire les coûts et à traiter les messages efficacement à l'aide d'Amazon SQS.

Rubriques

Utilisation des messages Amazon SQS

Les consignes suivantes peuvent vous aider à traiter les messages efficacement à l'aide d'Amazon SQS.

Rubriques

Traitement des messages en temps opportun

Le délai de visibilité défini dépend de la durée nécessaire à votre application pour traiter et supprimer un message. Par exemple, si votre application a besoin de 10 secondes pour traiter un message et que vous définissez un délai de visibilité de 15 minutes, vous devez attendre relativement longtemps pour tenter de retraiter le message si la dernière tentative de traitement échoue. En revanche, si votre application a besoin de 10 secondes pour traiter un message, mais que vous définissez un délai de visibilité de seulement 2 secondes, un message dupliqué est reçu par un autre consommateur alors que le consommateur d'origine est toujours en train de travailler sur le message.

Pour vérifier qu'il y a suffisamment de temps pour traiter les messages, utilisez l'une des stratégies suivantes :

- Si vous savez (ou si vous pouvez raisonnablement estimer) combien de temps est nécessaire pour traiter un message, rallongez le délai de visibilité de sorte qu'il corresponde à la durée maximale requise pour traiter et supprimer le message. Pour plus d'informations, consultez Configuration du délai de visibilité.
- Si vous ne savez pas combien de temps il faut pour traiter un message, créez une pulsation pour votre processus de consommateur : spécifiez le délai de visibilité initial (par exemple, 2 minutes) puis, tant que votre client travaille sur le message, continuez à prolonger le délai de visibilité de 2 minutes, toutes les minutes.

> **Important**
>
> Le délai maximal de visibilité est de 12 heures à compter de l'heure où Amazon SQS reçoit la demande `ReceiveMessage`. L'extension du délai d'attente de visibilité ne réinitialise pas le maximum de 12 heures.
> En outre, il se peut que vous ne puissiez pas régler le délai d'expiration d'un message individuel sur 12 heures (par exemple, 43 200 secondes) puisque la demande `ReceiveMessage` déclenche le temporisateur. Par exemple, si vous recevez un message et que vous définissez immédiatement le maximum de 12 heures en envoyant un appel `ChangeMessageVisibility` avec `VisibilityTimeout` défini sur une durée égale à 43 200 secondes, il échouera probablement. En revanche, l'utilisation d'une valeur de 43 195 secondes fonctionnera, à moins qu'il n'y ait un délai important entre la demande du message via `ReceiveMessage` et la mise à jour du délai de visibilité. Si votre client a besoin de plus de 12 heures, envisagez d'utiliser Step Functions.

Gestion des erreurs de demande

Pour gérer les erreurs de demande, utilisez l'une des stratégies suivantes :

- Si vous utilisez un AWS SDK, vous disposez déjà d'une logique de réessai et d'annulation automatiques. Pour plus d'informations, consultez Error Retries and Exponential Backoff in AWS(Tentatives sur l'erreur et backoff exponentiel)) dans Référence générale d'Amazon Web Services.
- Si vous n'utilisez pas les fonctionnalités du AWS SDK pour réessayer et annuler, attendez une pause (par exemple, 200 ms) avant de réessayer l'ReceiveMessageaction après n'avoir reçu aucun message, un délai d'expiration ou un message d'erreur d'Amazon SQS. Pour permettre une utilisation ultérieure de `ReceiveMessage` qui donne les mêmes résultats, patientez plus longtemps (par exemple, 400 ms).

Configuration de l'interrogation longue

Lorsque le temps d'attente pour l'action de l'API `ReceiveMessage` est supérieur à 0, une recherche prolongée est activée. Le temps d'attente maximal pour la recherche prolongée est de 20 secondes. La recherche prolongée permet de réduire le coût d'utilisation d'Amazon SQS en éliminant le nombre de réponses vides (lorsqu'il n'y a aucun message disponible pour une demande `ReceiveMessage`) et de fausses réponses vides (lorsque les messages sont disponibles dans la file d'attente, mais ne sont pas inclus dans une réponse). Pour plus d'informations, consultez Recherches courtes et longues sur Amazon SQS.

Pour garantir un traitement optimal des messages, utilisez les stratégies suivantes :

- Dans la plupart des cas, vous pouvez définir le délai d'attente `ReceiveMessage` à 20 secondes. Si un délai de 20 secondes est trop long pour votre application, définissez un délai d'attente `ReceiveMessage` plus court (1 seconde au minimum). Si vous n'utilisez pas de AWS SDK pour accéder à Amazon SQS, ou si vous configurez AWS un SDK pour réduire le temps d'attente, il se peut que vous deviez modifier votre client Amazon SQS pour autoriser des demandes plus longues ou utiliser un temps d'attente plus court pour les longs sondages.
- Si vous implémentez l'attente active de longue durée pour plusieurs files d'attente, utilisez un thread pour chacune d'elles plutôt qu'un seul thread pour toutes les files d'attente. L'utilisation d'un seul thread pour chaque file d'attente permet à votre application de traiter les messages de chacune des files d'attente dès qu'ils sont disponibles. En revanche, en utilisant un seul thread pour l'interrogation de plusieurs files d'attente peut mettre votre application dans l'incapacité de

traiter les messages disponibles dans d'autres files d'attente lorsque l'application attend (pendant une durée pouvant atteindre 20 secondes) la file d'attente qui ne comporte aucun message.

> **Important**
>
> Pour éviter les erreurs HTTP, assurez-vous que le temps d'attente de réponse HTTP pour les demandes `ReceiveMessage` est plus long que le paramètre `WaitTimeSeconds`. Pour plus d'informations, consultez ReceiveMessage.

Capture des messages problématiques

Pour capturer tous les messages qui ne peuvent pas être traités et pour collecter des CloudWatch statistiques précises, configurez une file d'attente de lettres mortes.

- La stratégie de redirection redirige les messages vers une file d'attente de lettres mortes lorsque la file d'attente source ne parvient pas à traiter un message après un nombre de tentatives spécifique.
- L'utilisation d'une file d'attente de lettres mortes limite le nombre de messages et réduit la possibilité d'être exposé à des messages de type « poison pill » (messages reçus mais ne pouvant pas être traités).
- L'inclusion d'un message de pilule empoisonnée dans une file d'attente peut fausser la `ApproximateAgeOfOldestMessage` CloudWatch métrique en indiquant un âge incorrect au message de pilule empoisonnée. La configuration d'une file d'attente de lettres mortes contribue à éviter les fausses alarmes lors de l'utilisation de cette métrique.

Configuration de la conservation des files d'attente de lettres mortes

Pour les files d'attente standard, l'expiration d'un message est toujours basée sur son horodatage de mise en file d'attente d'origine. Lorsqu'un message est déplacé vers une file d'attente de lettres mortes, l'horodatage de la mise en file d'attente reste inchangé. La métrique `ApproximateAgeOfOldestMessage` indique à quel moment le message a été placé dans la file d'attente de lettres mortes, et non à quel moment le message a été initialement envoyé. Supposons, par exemple, qu'un message passe 1 journée dans la file d'attente d'origine avant d'être déplacé vers une file d'attente de lettres mortes. Si la période de conservation de la file d'attente de lettres mortes est de 4 jours, le message est supprimé de la file d'attente de lettres mortes au bout de 3 jours et le paramètre `ApproximateAgeOfOldestMessage` est défini sur 3 jours. Il est donc recommandé de

toujours définir la période de rétention d'une file d'attente de lettres mortes de manière à ce qu'elle soit plus longue que la période de rétention de la file d'attente d'origine.

Pour les files d'attente FIFO, l'horodatage de la mise en file d'attente est réinitialisé lorsque le message est déplacé vers une file d'attente de lettres mortes. La métrique `ApproximateAgeOfOldestMessage` indique à quel moment le message a été placé dans la file d'attente de lettres mortes. Dans le même exemple ci-dessus, le message est supprimé de la file d'attente de lettres mortes au bout de 4 jours et le paramètre `ApproximateAgeOfOldestMessage` est défini sur 4 jours.

Éviter le traitement de message incohérent

Amazon SQS est un système distribué. Il est donc possible qu'un consommateur ne reçoive pas de message, même lorsque Amazon SQS marque le message comme remis lors d'un retour réussi à partir d'un appel de méthode d'API `ReceiveMessage`. Dans ce cas, Amazon SQS enregistre le message tel qu'il a été remis au moins une fois, bien que le consommateur ne l'ait jamais reçu. Étant donné qu'aucune tentative supplémentaire de remise de messages n'est effectuée dans ces conditions, nous ne recommandons pas de définir le nombre maximal de réceptions sur 1 pour une file d'attente de lettres mortes.

Implémentation de systèmes de demande-réponse

Lors de l'implémentation d'un système de demande-réponse ou d'appel de procédure éloigné (RPC), ayez les bonnes pratiques suivantes à l'esprit :

- Ne créez pas des files d'attente de réponses par message. Créez plutôt des files d'attente de réponses au démarrage, par producteur, et utilisez un attribut de message d'identification de corrélation pour mapper les réponses aux demandes.
- Ne permettez pas à vos producteurs de partager des files d'attente de réponses. Un producteur risquerait de recevoir des messages de réponse destinés à un autre producteur.

Pour plus d'informations sur l'implémentation du modèle demande-réponse à l'aide du client de file d'attente temporaire, consultez Modèle de messagerie demande-réponse (files d'attente virtuelles).

Réduire les coûts Amazon SQS

Les bonnes pratiques suivantes peuvent vous aider à réduire les coûts et à tirer parti d'une potentielle réduction des coûts supplémentaires et d'une réponse quasi-instantanée.

Actions de message par lots

Afin de réduire les coûts, regroupez vos actions de message :

- Pour envoyer, recevoir et supprimer des messages, et modifier le délai de visibilité de plusieurs messages en une seule opération, utilisez les actions d'API de traitement par lots d'Amazon SQS.
- Pour combiner la mise en tampon côté client avec le traitement par lots des demandes, utilisez l'attente active de longue durée avec le client asynchrone mis en tampon inclus dans le kit AWS SDK for Java.

> **Note**
>
> Le client asynchrone en mémoire tampon Amazon SQS ne prend actuellement pas en charge les files d'attente FIFO.

Utilisation du mode d'interrogation approprié

- La recherche prolongée vous permet de consommer des messages de votre file d'attente Amazon SQS dès qu'ils sont disponibles.
  - Pour réduire le coût d'utilisation d'Amazon SQS et le nombre de réceptions vides dans une file d'attente vide (réponses à l'action `ReceiveMessage` qui ne renvoient aucun message), activez la recherche prolongée. Pour plus d'informations, consultez Recherche prolongée Amazon SQS.
  - Pour accroître l'efficacité lors de l'attente active de plusieurs threads avec plusieurs réceptions, diminuez le nombre de threads.
  - Dans la plupart des cas, l'attente active de longue durée est préférable à celle de courte durée.
- L'attente active de courte durée retourne des réponses immédiatement, même si la file d'attente Amazon SQS interrogée est vide.
  - Pour satisfaire aux exigences d'une application qui attend des réponses immédiates à la demande `ReceiveMessage`, utilisez l'attente active de courte durée.
  - L'attente active de courte durée coûte le même prix que l'attente active de longue durée.

Passage d'une file d'attente standard à une file d'attente FIFO Amazon SQS

Si vous ne voulez pas régler le paramètre `DelaySeconds` sur chaque message, vous pouvez passer à une file d'attente FIFO en fournissant un ID de groupe de messagerie pour chaque message envoyé.

Pour plus d'informations, voir Passage d'une file d'attente standard à une file d'attente FIFO dans Amazon SQS.

Recommandations supplémentaires pour les files d'attente FIFO Amazon SQS

Les bonnes pratiques suivantes peuvent vous permettre d'utiliser l'ID de déduplication du message et l'ID de groupe de messages de façon optimale. Pour plus d'informations, consultez les actions `SendMessage` et `SendMessageBatch` dans la Référence d'API Amazon Simple Notification Service.

Rubriques

- Utilisation de l'ID de déduplication du message Amazon SQS
- Utilisation de l'ID de groupe de messagerie Amazon SQS
- Utilisation de l'ID de tentative de demande de réception Amazon SQS

Utilisation de l'ID de déduplication du message Amazon SQS

L'ID de déduplication de messages est le jeton utilisé pour la déduplication des messages envoyés. Si un message avec un ID de déduplication de message particulier est correctement envoyé, tous les messages envoyés avec le même ID de déduplication de message sont correctement acceptés, mais ne sont pas remis pendant l'intervalle de déduplication de 5 minutes.

> **Note**
>
> Amazon SQS continue de suivre l'ID de déduplication du message même après la réception et la suppression du message.

Fourniture de l'ID de déduplication du message

Le producteur doit fournir des valeurs d'ID de déduplication du message pour chaque message dans les scénarios suivants :

- Messages envoyés avec des corps identiques qu'Amazon SQS doit traiter comme uniques.
- Messages envoyés avec un contenu identique, mais des attributs différents qu'Amazon SQS doit traiter comme uniques.
- Messages envoyés avec un contenu différent (par exemple, le nombre de tentatives inclus dans le corps du message) qu'Amazon SQS doit traiter comme des doublons.

Activation de la déduplication pour un système producteur/consommateur unique

Si vous avez un seul producteur et un seul consommateur, et que les messages sont uniques parce qu'un ID de message spécifique à l'application est inclus dans le corps du message, suivez ces bonnes pratiques :

- Activez la déduplication basée sur le contenu pour la file d'attente (chacun de vos messages a un corps unique). Le producteur peut ignorer l'ID de déduplication du message.
- Lorsque la déduplication basée sur le contenu est activée pour une file d'attente Amazon SQS FIFO et qu'un message est envoyé avec un ID de déduplication, l'ID de déduplication remplace l'ID de déduplication basé sur le contenu `SendMessage` généré.
- Même si le consommateur n'est pas tenu de fournir un ID de tentative de demande de réception pour chaque demande, il vaut mieux le faire, car cela permet aux séquences échec-réessayer de s'exécuter plus rapidement.
- Vous pouvez réessayer d'envoyer ou de recevoir des demandes, car elles n'interfèrent pas avec l'ordre des messages dans les files d'attente FIFO.

Conception de scénarios de récupération après une panne

Le processus de déduplication dans les files d'attente FIFO est prioritaire. Lors de la conception de l'application, assurez-vous que le producteur et le consommateur peuvent récupérer en cas de panne du client ou du réseau.

- Le producteur doit connaître l'intervalle de déduplication de la file d'attente. Amazon SQS a un intervalle de déduplication de 5 minutes. De nouvelles tentatives de demandes de `SendMessage`

après expiration du délai de déduplication peuvent introduire des messages en double dans la file d'attente. Par exemple, un appareil mobile dans une voiture envoie des messages dont l'ordre est important. Si la voiture perd la connectivité cellulaire pendant un certain temps avant de recevoir une confirmation, une nouvelle tentative de la demande après la récupération de la connectivité cellulaire peut créer un doublon.

- Le consommateur doit disposer d'un délai de visibilité réduisant le risque de ne pas pouvoir traiter des messages avant l'expiration de ce délai. Vous pouvez étendre le délai de visibilité pendant que les messages sont en cours de traitement en appelant l'action `ChangeMessageVisibility`. Toutefois, si le délai de visibilité expire, un autre consommateur peut immédiatement commencer à traiter les messages, un message étant alors traité plusieurs fois. Pour éviter ce scénario, configurez une file d'attente de lettres mortes.

Utilisation des délais de visibilité

Pour des performances optimales, définissez le délai de visibilité pour qu'il soit supérieur au délai de lecture du AWS SDK. Cela s'applique à l'utilisation de l'action d'API `ReceiveMessage` avec l'attente active de courte durée ou l'attente active de longue durée.

Utilisation de l'ID de groupe de messagerie Amazon SQS

`MessageGroupId` est la balise qui spécifie qu'un message appartient à un groupe de messages spécifique. Les messages appartenant au même groupe de messages sont toujours traités un à la fois, dans un ordre strict par rapport au groupe de messages (toutefois, les messages appartenant à des groupes de messages différents peuvent être traités dans le désordre).

Entrelacement de plusieurs groupes de messages classés

Pour entrelacer plusieurs groupes de messages classés au sein d'une seule file d'attente FIFO, utilisez les valeurs d'ID de groupe de messages (par exemple, les données de session de plusieurs utilisateurs). Dans ce scénario, plusieurs consommateurs peuvent traiter la file d'attente, mais les données de session de chaque utilisateur sont traitées selon la procédure FIFO.

> **Note**
>
> Lorsque les messages qui appartiennent à un ID de groupe de messages particulier sont invisibles, aucun autre consommateur ne peut traiter les messages avec le même ID de groupe de messages.

Éviter de traiter les doublons dans un système avec plusieurs producteurs/consommateurs

Pour éviter de traiter les messages en double dans un système avec plusieurs producteurs et consommateurs où le débit et la latence sont plus importants que le classement, le producteur doit générer un ID de groupe de messages unique pour chaque message.

> **Note**
>
> Dans ce scénario, les doubles sont supprimés. Toutefois, le classement des messages ne peut pas être garanti.
> Tout scénario avec plusieurs producteurs et consommateurs augmente le risque de diffuser par inadvertance un message dupliqué, si aucun employé ne traite le message pendant le délai de visibilité et que celui devient accessible à un autre employé.

Éviter d'avoir un nombre important de messages en attente avec le même ID de groupe de messages

Pour les files d'attente FIFO, il peut y avoir un maximum de 20 000 messages en cours (reçus depuis une file d'attente par un consommateur, mais pas encore supprimés de la file d'attente). Si vous atteignez ce quota, Amazon SQS ne renvoie aucun message d'erreur. Une file d'attente FIFO examine les 20 000 premiers messages pour déterminer les groupes de messages disponibles. Cela signifie que si vous avez un arriéré de messages dans un seul groupe de messages, vous ne pouvez pas consommer les messages d'autres groupes de messages envoyés à la file d'attente ultérieurement tant que vous n'avez pas correctement consommé les messages du backlog.

> **Note**
>
> Des messages en attente ayant le même ID de groupe de messages peuvent s'accumuler à cause d'un consommateur qui ne parvient pas à traiter un message. Des problèmes de traitement des messages peuvent se produire en raison d'un problème lié au contenu d'un message ou d'un problème technique avec le consommateur.
> Pour déplacer les messages qui ne peuvent pas être traités de manière répétée et pour débloquer le traitement des autres messages ayant le même ID de groupe de messages, envisagez de configurer une stratégie de file d'attente de lettres mortes.

Évitez de réutiliser le même ID de groupe de messages avec des files d'attente virtuelles

Pour éviter que les messages ayant le même ID de groupe de messages envoyés à différentes files d'attente virtuelles avec la même file d'attente d'hôte ne se bloquent mutuellement, évitez de réutiliser le même ID de groupe de messages avec des files d'attente virtuelles.

Utilisation de l'ID de tentative de demande de réception Amazon SQS

L'ID de tentative de demande de réception est le jeton utilisé pour la déduplication des appels `ReceiveMessage`.

Pendant une longue indisponibilité du réseau qui entraîne des problèmes de connectivité entre le kit SDK et Amazon SQS, il est recommandé de fournir l'ID de tentative de demande de réception et de réessayer avec le même ID de tentative de demande de réception en cas d'échec de l'opération du kit SDK.

Exemples de SDK Java Amazon SQS

Vous pouvez les utiliser AWS SDK for Java pour créer des applications Java qui interagissent avec Amazon Simple Queue Service (Amazon SQS) et d'autres services. AWS Pour installer et configurer le SDK, reportez-vous à la section Premiers pas du Guide du développeur AWS SDK for Java 2.x .

Pour obtenir des exemples d'opérations de base sur les files d'attente Amazon SQS, telles que la création d'une file d'attente ou l'envoi d'un message, consultez la section Utilisation des files d'attente de messages Amazon SQS dans le Guide du développeur AWS SDK for Java 2.x .

Les exemples présentés dans cette rubrique présentent des fonctionnalités supplémentaires d'Amazon SQS, telles que le chiffrement côté serveur (SSE), les balises de répartition des coûts et les attributs des messages.

Rubriques

Utilisation du chiffrement côté serveur avec les files d'attente Amazon SQS

Vous pouvez utiliser le AWS SDK for Java pour ajouter un chiffrement côté serveur (SSE) à une file d'attente Amazon SQS. Chaque file d'attente utilise une clé KMS AWS Key Management Service (AWS KMS) pour générer les clés de chiffrement des données. Cet exemple utilise la clé KMS AWS gérée pour Amazon SQS. Pour en savoir plus sur l'utilisation du SSE et sur le rôle de la clé KMS, consultez Chiffrement au repos dans Amazon SQS.

Ajout du SSE à une file d'attente existante

Pour activer le chiffrement côté serveur pour une file d'attente existante, utilisez la méthode `SetQueueAttributes` afin de définir l'attribut `KmsMasterKeyId`.

L'exemple de code suivant définit la clé KMS AWS KMS key comme clé KMS AWS gérée pour Amazon SQS. L'exemple définit également la période de réutilisation de la AWS KMS key sur 140 secondes.

Avant d'exécuter l'exemple de code, assurez-vous d'avoir défini vos AWS informations d'identification. Pour plus d'informations, consultez la section Configurer les AWS informations d'identification et la région pour le développement dans le guide du AWS SDK for Java 2.x développeur.

```
// Create an SqsClient for the specified Region.
SqsClient sqsClient = SqsClient.builder().region(Region.US_WEST_1).build();

// Get the URL of your queue.
String myQueueName = "my queue";
GetQueueUrlResponse getQueueUrlResponse =

 sqsClient.getQueueUrl(GetQueueUrlRequest.builder().queueName(myQueueName).build());
String queueUrl = getQueueUrlResponse.queueUrl();

// Create a hashmap for the attributes. Add the key alias and reuse period to the
 hashmap.
HashMap<QueueAttributeName, String> attributes = new HashMap<QueueAttributeName,
 String>();
final String kmsMasterKeyAlias = "alias/aws/sqs";  // the alias of the AWS managed KMS
 key for Amazon SQS.
attributes.put(QueueAttributeName.KMS_MASTER_KEY_ID, kmsMasterKeyAlias);
attributes.put(QueueAttributeName.KMS_DATA_KEY_REUSE_PERIOD_SECONDS, "140");

// Create the SetQueueAttributesRequest.
SetQueueAttributesRequest set_attrs_request = SetQueueAttributesRequest.builder()
          .queueUrl(queueUrl)
          .attributes(attributes)
          .build();

sqsClient.setQueueAttributes(set_attrs_request);
```

Désactivation du SSE pour une file d'attente

Pour désactiver le chiffrement côté serveur pour une file d'attente existante, définissez l'attribut `KmsMasterKeyId` sur une chaîne vide à l'aide de la méthode `SetQueueAttributes`.

> **Important**
>
> `null` n'est pas une valeur valide pour `KmsMasterKeyId`.

Création d'une file d'attente avec le SSE

Pour activer le SSE lorsque vous créez la file d'attente, ajoutez l'attribut `KmsMasterKeyId` à la méthode d'API `CreateQueue`.

L'exemple suivant crée une file d'attente avec le SSE activé. La file d'attente utilise la clé KMS gérée par AWS pour Amazon SQS. L'exemple définit également la période de réutilisation de la AWS KMS key sur 160 secondes.

Avant d'exécuter l'exemple de code, assurez-vous d'avoir défini vos AWS informations d'identification. Pour plus d'informations, consultez la section Configurer les AWS informations d'identification et la région pour le développement dans le guide du AWS SDK for Java 2.x développeur.

```
// Create an SqsClient for the specified Region.
SqsClient sqsClient = SqsClient.builder().region(Region.US_WEST_1).build();

// Create a hashmap for the attributes. Add the key alias and reuse period to the
 hashmap.
HashMap<QueueAttributeName, String> attributes = new HashMap<QueueAttributeName,
 String>();
final String kmsMasterKeyAlias = "alias/aws/sqs";  // the alias of the AWS managed KMS
 key for Amazon SQS.
attributes.put(QueueAttributeName.KMS_MASTER_KEY_ID, kmsMasterKeyAlias);
attributes.put(QueueAttributeName.KMS_DATA_KEY_REUSE_PERIOD_SECONDS, "140");

// Add the attributes to the CreateQueueRequest.
CreateQueueRequest createQueueRequest =
                CreateQueueRequest.builder()
                        .queueName(queueName)
                        .attributes(attributes)
                        .build();
sqsClient.createQueue(createQueueRequest);
```

Récupération des attributs SSE

Pour plus d'informations sur la récupération des attributs de file d'attente, consultez les Exemples de la Référence d'API Amazon Simple Queue Service.

Pour récupérer l'ID de la clé KMS ou la période de réutilisation de la clé de données pour une file d'attente particulière, exécutez la méthode `GetQueueAttributes` et récupérez les valeurs `KmsMasterKeyId` et `KmsDataKeyReusePeriodSeconds`.

Configuration des balises pour une file d'attente Amazon SQS

Utilisez des balises de répartition des coûts afin d'organiser et d'identifier vos files d'attente Amazon SQS. Les exemples suivants montrent comment configurer des balises à l'aide d' AWS SDK for Java. Pour plus d'informations, consultez Balises de répartition des coûts Amazon SQS.

Avant d'exécuter l'exemple de code, assurez-vous d'avoir défini vos AWS informations d'identification. Pour plus d'informations, consultez la section Configurer les AWS informations d'identification et la région pour le développement dans le guide du AWS SDK for Java 2.x développeur.

Établissement d'une liste de balises

Pour répertorier les balises d'une file d'attente, utilisez la méthode `ListQueueTags`.

```
// Create an SqsClient for the specified region.
SqsClient sqsClient = SqsClient.builder().region(Region.US_WEST_1).build();

// Get the queue URL.
String queueName = "MyStandardQ1";
GetQueueUrlResponse getQueueUrlResponse =

 sqsClient.getQueueUrl(GetQueueUrlRequest.builder().queueName(queueName).build());
String queueUrl = getQueueUrlResponse.queueUrl();

// Create the ListQueueTagsRequest.
final ListQueueTagsRequest listQueueTagsRequest =

 ListQueueTagsRequest.builder().queueUrl(queueUrl).build();

// Retrieve the list of queue tags and print them.
final ListQueueTagsResponse listQueueTagsResponse =
                                  sqsClient.listQueueTags(listQueueTagsRequest);
System.out.println(String.format("ListQueueTags: \tTags for queue %s are %s.\n",
                queueName, listQueueTagsResponse.tags() ));
```

Ajout ou mise à jour de balises

Pour ajouter ou mettre à jour les valeurs de balise d'une file d'attente, utilisez la méthode `TagQueue`.

```
 // Create an SqsClient for the specified Region.
SqsClient sqsClient = SqsClient.builder().region(Region.US_WEST_1).build();

// Get the queue URL.
String queueName = "MyStandardQ1";
GetQueueUrlResponse getQueueUrlResponse =

 sqsClient.getQueueUrl(GetQueueUrlRequest.builder().queueName(queueName).build());
String queueUrl = getQueueUrlResponse.queueUrl();

// Build a hashmap of the tags.
final HashMap<String, String> addedTags = new HashMap<>();
        addedTags.put("Team", "Development");
        addedTags.put("Priority", "Beta");
        addedTags.put("Accounting ID", "456def");

//Create the TagQueueRequest and add them to the queue.
final TagQueueRequest tagQueueRequest = TagQueueRequest.builder()
        .queueUrl(queueUrl)
        .tags(addedTags)
        .build();
sqsClient.tagQueue(tagQueueRequest);
```

Suppression de balises

Pour supprimer une ou plusieurs balises de la file d'attente, utilisez la méthode `UntagQueue`. L'exemple suivant supprime la balise `Accounting ID`.

```

// Create the UntagQueueRequest.
final UntagQueueRequest untagQueueRequest = UntagQueueRequest.builder()
        .queueUrl(queueUrl)
        .tagKeys("Accounting ID")
        .build();

// Remove the tag from this queue.
sqsClient.untagQueue(untagQueueRequest);
```

Envoi d'attributs de message à une file d'attente Amazon SQS

Vous pouvez inclure des métadonnées structurées (telles que des horodatages, des données géospatiales, des signatures et des identifiants) dans des messages utilisant des attributs de message. Pour plus d'informations, consultez Attributs de message Amazon SQS.

Avant d'exécuter l'exemple de code, assurez-vous d'avoir défini vos AWS informations d'identification. Pour plus d'informations, consultez la section Configurer les AWS informations d'identification et la région pour le développement dans le guide du AWS SDK for Java 2.x développeur.

Définition des attributs

Pour définir un attribut pour un message, ajoutez le code suivant qui utilise le type de données `MessageAttributeValue`. Pour plus d'informations, consultez Composants des attributs de message et Types de données d'attribut de message.

calcule AWS SDK for Java automatiquement les sommes de contrôle du corps et des attributs du message et les compare aux données renvoyées par Amazon SQS. Pour plus d'informations, consultez le Guide du développeur AWS SDK for Java 2.x et Calcul de la valeur de hachage MD5 pour les attributs de message pour les autres langages de programmation.

String

Cet exemple définit un attribut `String` nommé `Name` avec la valeur `Jane`.

```
final Map<String, MessageAttributeValue> messageAttributes = new HashMap<>();
messageAttributes.put("Name", new MessageAttributeValue()
.withDataType("String")
.withStringValue("Jane"));
```

Number

Cet exemple définit un attribut `Number` nommé `AccurateWeight` avec la valeur `230.0000000000000001`.

```
final Map<String, MessageAttributeValue> messageAttributes = new HashMap<>();
```

```
messageAttributes.put("AccurateWeight", new MessageAttributeValue()
.withDataType("Number")
.withStringValue("230.00000000000000001"));
```

Binary

Cet exemple définit un attribut `Binary` nommé `ByteArray` avec la valeur d'une matrice de 10 octets non initialisée.

```
final Map<String, MessageAttributeValue> messageAttributes = new HashMap<>();
messageAttributes.put("ByteArray", new MessageAttributeValue()
.withDataType("Binary")
.withBinaryValue(ByteBuffer.wrap(new byte[10])));
```

String (custom)

Cet exemple définit l'attribut personnalisé `String.EmployeeId` nommé `EmployeeId` avec la valeur `ABC123456`.

```
final Map<String, MessageAttributeValue> messageAttributes = new HashMap<>();
messageAttributes.put("EmployeeId", new MessageAttributeValue()
.withDataType("String.EmployeeId")
.withStringValue("ABC123456"));
```

Number (custom)

Cet exemple définit l'attribut personnalisé `Number.AccountId` nommé `AccountId` avec la valeur `000123456`.

```
final Map<String, MessageAttributeValue> messageAttributes = new HashMap<>();
messageAttributes.put("AccountId", new MessageAttributeValue()
.withDataType("Number.AccountId")
.withStringValue("000123456"));
```

> **Note**
>
> Comme le type de données de base est `Number`, la méthode `ReceiveMessage` renvoie `123456`.

Binary (custom)

Cet exemple définit l'attribut personnalisé `Binary.JPEG` nommé `ApplicationIcon` avec la valeur d'une matrice de 10 octets non initialisée.

```
final Map<String, MessageAttributeValue> messageAttributes = new HashMap<>();
messageAttributes.put("ApplicationIcon", new MessageAttributeValue()
.withDataType("Binary.JPEG")
.withBinaryValue(ByteBuffer.wrap(new byte[10])));
```

Envoi d'un message avec des attributs

Cet exemple ajoute les attributs à `SendMessageRequest` avant d'envoyer le message.

```
// Send a message with an attribute.
final SendMessageRequest sendMessageRequest = new SendMessageRequest();
sendMessageRequest.withMessageBody("This is my message text.");
sendMessageRequest.withQueueUrl(myQueueUrl);
sendMessageRequest.withMessageAttributes(messageAttributes);
sqs.sendMessage(sendMessageRequest);
```

Important

Si vous envoyez un message à une file d'attente FIFO (First-In First-Out), assurez-vous que la méthode `sendMessage` s'exécute après que vous avez fourni l'ID de groupe de messages.
Si vous utilisez la méthode `SendMessageBatch` au lieu de `SendMessage`, vous devez spécifier les attributs de message pour chaque message du lot.

Utilisation des API Amazon SQS

Cette section fournit des informations sur la création de points de terminaison Amazon SQS et de demandes d'API de requête à l'aide des méthodes GET et POST, et l'utilisation des actions d'API groupées. Pour obtenir des informations détaillées sur les actions Amazon SQS, notamment les paramètres, les erreurs, les exemples et les types de données, consultez la Référence d'API Amazon Simple Queue Service.

Pour accéder à Amazon SQS via différents langages de programmation, vous pouvez également utiliser les kits SDK AWS qui contiennent les fonctionnalités automatiques suivantes :

- Signature cryptographique des requêtes de service
- Nouvelles tentatives de requête
- Gestion des réponses d'erreur

Pour plus d'informations sur l'outil de ligne de commande, consultez les sections Amazon SQS dans la Référence de commande AWS CLI et la Référence Cmdlet AWS Tools for PowerShell .

API Amazon SQS avec AWS protocole JSON

Amazon SQS utilise le protocole AWS JSON comme mécanisme de transport pour toutes les API Amazon SQS sur les versions du AWS SDK spécifiées. AWS Le protocole JSON fournit un débit plus élevé, une latence plus faible et une application-to-application communication plus rapide. AWS Le protocole JSON est plus efficace dans la sérialisation/désérialisation des demandes et des réponses que le protocole de requête. AWS Si vous préférez toujours utiliser le protocole de AWS requête avec les API SQS, consultez Quels sont les langages pris en charge pour le protocole AWS JSON utilisé dans les API Amazon SQS ? les versions du AWS SDK qui prennent en charge le protocole de requête Amazon AWS SQS.

Amazon SQS utilise le protocole AWS JSON pour communiquer entre les clients du AWS SDK (par exemple, Java, Python, Golang JavaScript) et le serveur Amazon SQS. Une requête HTTP d'une opération d'API Amazon SQS accepte une entrée au format JSON. L'opération Amazon SQS est exécutée et la réponse d'exécution est renvoyée au client du SDK au format JSON. Comparé à la AWS requête, le AWS JSON est plus simple, plus rapide et plus efficace pour transporter les données entre le client et le serveur.

- AWS Le protocole JSON joue le rôle de médiateur entre le client et le serveur Amazon SQS.

- Le serveur ne comprend pas le langage de programmation dans lequel l'opération Amazon SQS est créée, mais il comprend le protocole AWS JSON.
- Le protocole AWS JSON utilise la sérialisation (conversion de l'objet au format JSON) et la désérialisation (conversion du format JSON en objet) entre le client et le serveur Amazon SQS.

Pour plus d'informations sur le protocole AWS JSON avec Amazon SQS, consultez. FAQ sur le protocole Amazon SQS AWS JSON

AWS Le protocole JSON est disponible sur la version du AWS SDK spécifiée. Pour consulter la version et les dates de sortie du kit SDK selon les variantes linguistiques, consultez la Matrice de prise en charge des versions des kits SDK et des outils AWS dans le Guide de référence des kits SDK et des outils AWS .

Rubriques

- Effectuer des demandes d'API de requête à l'aide du protocole AWS JSON dans Amazon SQS
- Effectuer des demandes d'API de requête à l'aide du protocole de AWS requête dans Amazon SQS
- Authentification des demandes pour Amazon SQS
- Actions groupées Amazon SQS

Effectuer des demandes d'API de requête à l'aide du protocole AWS JSON dans Amazon SQS

Dans cette section, vous apprenez à construire un point de terminaison Amazon SQS, à créer des requêtes POST et à interpréter les réponses.

> **Note**
>
> AWS Le protocole JSON est pris en charge pour la plupart des variantes linguistiques. Pour accéder à la liste complète des langages pris en charge, consultez Quels sont les langages pris en charge pour le protocole AWS JSON utilisé dans les API Amazon SQS ?.

Rubriques

- Constitution d'un point de terminaison

- Envoi de requête POST
- Interprétation des réponses de l'API JSON Amazon SQS
- FAQ sur le protocole Amazon SQS AWS JSON

Constitution d'un point de terminaison

Afin d'utiliser des files d'attente Amazon SQS, vous devez construire un point de terminaison. Pour plus d'informations sur les points de terminaison Amazon SQS, consultez les pages suivantes dans le Référence générale d'Amazon Web Services :

- Points de terminaison régionaux
- Points de terminaison et quotas Amazon Simple Queue Service

Chaque point de terminaison Amazon SQS est indépendant. Par exemple, si deux files d'attente sont nommées MyQueueet que l'une contient le point de terminaison `sqs.us-east-2.amazonaws.com` tandis que l'autre possède le point de terminaison`sqs.eu-west-2.amazonaws.com`, les deux files d'attente ne partagent aucune donnée entre elles.

L'exemple suivant correspond à un point de terminaison lançant une requête pour créer une file d'attente.

```
POST / HTTP/1.1
Host: sqs.us-west-2.amazonaws.com
X-Amz-Target: AmazonSQS.CreateQueue
X-Amz-Date: <Date>
Content-Type: application/x-amz-json-1.0
Authorization: <AuthParams>
Content-Length: <PayloadSizeBytes>
Connection: Keep-Alive
{
    "QueueName":"MyQueue",
    "Attributes": {
        "VisibilityTimeout": "40"
    },
    "tags": {
        "QueueType": "Production"
    }
}
```

> **Note**
>
> Les noms et les URL des files d'attente sont sensibles à la casse.
> La structure de *AUTHPARAMS* dépend de la signature de la demande d'API. Pour plus d'informations, consultez Signing AWS API Requests dans le manuel Amazon Web Services General Reference.

Envoi de requête POST

Une demande POST Amazon SQS envoie des paramètres de requête sous forme de formulaire dans le corps d'une demande HTTP.

Voici un exemple d'en-tête HTTP avec `X-Amz-Target` défini sur `AmazonSQS.<operationName>` et d'en-tête HTTP avec `Content-Type` défini sur `application/x-amz-json-1.0`.

```
POST / HTTP/1.1
Host: sqs.<region>.<domain>
X-Amz-Target: AmazonSQS.SendMessage
X-Amz-Date: <Date>
Content-Type: application/x-amz-json-1.0
Authorization: <AuthParams>
Content-Length: <PayloadSizeBytes>
Connection: Keep-Alive
{
    "QueueUrl": "https://sqs.<region>.<domain>/<awsAccountId>/<queueName>/",
    "MessageBody": "This is a test message",
}
```

Cette demande POST HTTP envoie un message à une file d'attente Amazon SQS.

> **Note**
>
> Les deux en-têtes HTTP `X-Amz-Target` et `Content-Type` sont obligatoires.
> Votre client HTTP peut ajouter d'autres éléments à la requête HTTP, en fonction de la version HTTP du client.

Interprétation des réponses de l'API JSON Amazon SQS

En réponse à une demande d'action, Amazon SQS renvoie une structure de données JSON qui contient les résultats de la demande. Pour plus d'informations, consultez les actions individuelles dans la Référence d'API Amazon Simple Queue Service et dans la FAQ sur le protocole Amazon SQS AWS JSON.

Rubriques

Structure d'une réponse JSON positive

Si la demande aboutit, l'élément de réponse principal est `x-amzn-RequestId`, qui contient l'identifiant unique universel (UUID) de la demande, ainsi que d'autres champs de réponse ajoutés. Par exemple, la réponse `CreateQueue` suivante contient le champ `QueueUrl`, qui contient à son tour l'URL de la file d'attente créée.

```
HTTP/1.1 200 OK
x-amzn-RequestId: <requestId>
Content-Length: <PayloadSizeBytes>
Date: <Date>
Content-Type: application/x-amz-json-1.0
{
    "QueueUrl":"https://sqs.us-east-1.amazonaws.com/111122223333/MyQueue"
}
```

Structure d'une réponse d'erreur JSON

Si une demande échoue, Amazon SQS renvoie la réponse principale, y compris l'en-tête HTTP et le corps du message.

Dans l'en-tête HTTP, `x-amzn-RequestId` contient l'UUID de la demande. `x-amzn-query-error` contient deux informations : le type d'erreur et s'il s'agit d'une erreur du producteur ou du consommateur.

Dans le corps de la réponse, `"__type"` indique les autres détails de l'erreur et `Message` indique la condition d'erreur dans un format lisible.

Vous trouverez ci-dessous un exemple de réponse d'erreur au format JSON :

```
HTTP/1.1 400 Bad Request
x-amzn-RequestId: 66916324-67ca-54bb-a410-3f567a7a0571
x-amzn-query-error: AWS.SimpleQueueService.NonExistentQueue;Sender
Content-Length: <PayloadSizeBytes>
Date: <Date>
Content-Type: application/x-amz-json-1.0
{
    "__type": "com.amazonaws.sqs#QueueDoesNotExist",
    "message": "The specified queue does not exist."
}
```

FAQ sur le protocole Amazon SQS AWS JSON

Questions fréquemment posées sur l'utilisation du protocole AWS JSON avec Amazon SQS.

Qu'est-ce que le protocole AWS JSON et en quoi diffère-t-il des demandes et réponses d'API Amazon SQS existantes ?

JSON est l'une des méthodes de connexion les plus utilisées et acceptées pour la communication entre des systèmes hétérogènes. Amazon SQS utilise le JSON comme moyen de communication entre un client AWS SDK (par exemple, Java, Python, Golang) et le serveur JavaScript Amazon SQS. La requête HTTP d'une opération d'API Amazon SQS accepte une entrée sous forme de JSON. L'opération Amazon SQS est exécutée et la réponse d'exécution est partagée avec le client du SDK sous forme de JSON. Comparé à la requête AWS , JSON est plus efficace pour transporter les données entre le client et le serveur.

- Le protocole Amazon SQS AWS JSON joue le rôle de médiateur entre le client et le serveur Amazon SQS.
- Le serveur ne comprend pas le langage de programmation dans lequel l'opération Amazon SQS est créée, mais il comprend le protocole AWS JSON.
- Le protocole Amazon SQS AWS JSON utilise la sérialisation (conversion de l'objet au format JSON) et la désérialisation (conversion du format JSON en objet) entre le client et le serveur Amazon SQS.

Comment démarrer avec les protocoles AWS JSON pour Amazon SQS ?

Pour commencer à utiliser la dernière version du AWS SDK afin d'accélérer la messagerie pour Amazon SQS, mettez à niveau AWS votre SDK vers la version spécifiée ou vers une version

ultérieure. Pour en savoir plus sur les clients SDK, consultez la colonne Guide dans le tableau ci-dessous.

Voici une liste des versions du SDK dans les variantes linguistiques du protocole AWS JSON à utiliser avec les API Amazon SQS :

Langue	Référentiel client SDK	Version du client SDK requise	Guide
C++	aws/aws-sdk-cpp	1,11,98	AWS SDK pour C++
Golang 1.x	aws/aws-sdk-go	v1.47.7	AWS SDK pour Go
Golang 2.x	aws/aws-sdk-go-v2	v1.28.0	AWS SDK pour Go V2
Java 1.x	aws/aws-sdk-java	1,12,585	AWS SDK pour Java
Java 2.x	aws/aws-sdk-java-v2	2,21,19	AWS SDK pour Java
JavaScript v2.x	aws/aws-sdk-js	v2.1492.0	JavaScript sur AWS
JavaScript v3.x	aws/aws-sdk-js-v3	v3.447.0	JavaScript sur AWS
.NET	aws/aws-sdk-net	3,7,681,0	AWS SDK pour .NET
PHP	aws/aws-sdk-php	3,285,2	AWS SDK pour PHP
Python-boto3	boto/boto3	1,28,82	AWS SDK pour Python (Boto3)
Python-botocore	boto/botocore	1,31,82	

Langue	Référentiel client SDK	Version du client SDK requise	Guide
			AWS SDK pour Python (Boto3)
awscli	AWS CLI	1,29,82	Interface de ligne de commande AWS
Ruby	aws/aws-sdk-ruby	1,67,0	AWS SDK pour Ruby

Quels sont les risques liés à l'activation du protocole JSON pour mes charges de travail Amazon SQS ?

Si vous utilisez une implémentation personnalisée du AWS SDK ou une combinaison de clients personnalisés et d'un AWS SDK pour interagir avec Amazon SQS qui AWS génère des réponses basées sur des requêtes (également appelées XML), cela peut être incompatible avec le protocole JSON. AWS Si vous rencontrez des problèmes, contactez le AWS Support.

Et si j'utilise déjà la dernière version du AWS SDK, mais que ma solution open source ne prend pas en charge le format JSON ?

Vous devez remplacer la version de votre kit SDK par la version antérieure à celle que vous utilisez. Voir Comment démarrer avec les protocoles AWS JSON pour Amazon SQS ? pour plus d'informations. AWS Les versions du SDK répertoriées dans Comment démarrer avec les protocoles AWS JSON pour Amazon SQS ? utilisent le protocole filaire JSON pour les API Amazon SQS. Si vous remplacez votre AWS SDK par la version précédente, vos API Amazon SQS AWS utiliseront la requête.

Quels sont les langages pris en charge pour le protocole AWS JSON utilisé dans les API Amazon SQS ?

Amazon SQS prend en charge toutes les variantes linguistiques pour lesquelles les AWS SDK sont généralement disponibles (GA). Actuellement, nous ne prenons pas en charge Kotlin, Rust ou Swift. Pour en savoir plus sur les autres variantes de langage, consultez la section Outils pour créer sur AWS.

Quelles sont les régions prises en charge pour le protocole AWS JSON utilisé dans les API Amazon SQS ?

Amazon SQS prend en charge le protocole AWS JSON dans toutes les AWS régions où Amazon SQS est disponible.

À quelles améliorations de latence puis-je m'attendre lors de la mise à niveau vers les versions du AWS SDK spécifiées pour Amazon SQS à l'aide AWS du protocole JSON ?

AWS Le protocole JSON est plus efficace pour la sérialisation et la désérialisation des demandes et des réponses que le protocole de AWS requête. Basé sur des tests de AWS performance pour une charge utile de messages de 5 Ko, le protocole JSON pour Amazon SQS end-to-end réduit la latence de traitement des messages jusqu'à 23 %, ainsi que l'utilisation du processeur et de la mémoire côté client de l'application.

Le protocole de AWS requête sera-t-il obsolète ?

AWS le protocole de requête continuera d'être pris en charge. Vous pouvez continuer à utiliser le protocole de AWS requête tant que la version de votre AWS SDK est définie sur une version précédente autre que celle répertoriée dans Comment démarrer avec les protocoles AWS JSON pour Amazon SQS.

Où puis-je trouver plus d'informations sur le protocole AWS JSON ?

Vous trouverez plus d'informations sur le protocole JSON dans Protocole AWS JSON 1.0 dans la documentation de Smithy. Pour en savoir plus sur les requêtes d'API Amazon SQS avec le protocole AWS JSON, consultez Effectuer des demandes d'API de requête à l'aide du protocole AWS JSON dans Amazon SQS.

Effectuer des demandes d'API de requête à l'aide du protocole de AWS requête dans Amazon SQS

Dans cette section, vous apprenez à construire un point de terminaison Amazon SQS, à créer des requêtes GET et POST, et à interpréter les réponses.

Rubriques

- Constitution d'un point de terminaison
- Envoi de requête GET
- Envoi de requête POST
- Interprétation des réponses de l'API XML Amazon SQS

Constitution d'un point de terminaison

Afin d'utiliser des files d'attente Amazon SQS, vous devez construire un point de terminaison. Pour plus d'informations sur les points de terminaison Amazon SQS, consultez les pages suivantes dans la Référence générale d'Amazon Web Services :

- Points de terminaison régionaux
- Points de terminaison et quotas Amazon Simple Queue Service

Chaque point de terminaison Amazon SQS est indépendant. Par exemple, si deux files d'attente sont nommées MyQueueet que l'une contient le point de terminaison `sqs.us-east-2.amazonaws.com` tandis que l'autre possède le point de terminaison`sqs.eu-west-2.amazonaws.com`, les deux files d'attente ne partagent aucune donnée entre elles.

L'exemple suivant correspond à un point de terminaison lançant une requête pour créer une file d'attente.

```
https://sqs.eu-west-2.amazonaws.com/
?Action=CreateQueue
&DefaultVisibilityTimeout=40
&QueueName=MyQueue
&Version=2012-11-05
&AUTHPARAMS
```

> **Note**
>
> Les noms et les URL des files d'attente sont sensibles à la casse.
> La structure de *AUTHPARAMS* dépend de la signature de la demande d'API. Pour plus d'informations, consultez Signing AWS API Requests dans le manuel Amazon Web Services General Reference.

Envoi de requête GET

Une requête GET Amazon SQS est structurée comme une URL se composant des éléments suivants :

- Point de terminaison : ressource sur laquelle la requête agit (nom de la file d'attente et URL), par exemple : `https://sqs.us-east-2.amazonaws.com/123456789012/MyQueue`
- Action : action que vous souhaitez effectuer sur le point de terminaison. Un point d'interrogation (?) sépare le point de terminaison de l'action, par exemple : `?Action=SendMessage&MessageBody=Your%20Message%20Text`.
- Paramètres : tout paramètre de demande. Chaque paramètre est séparé par une esperluette (&), par exemple : `&Version=2012-11-05&AUTHPARAMS`.

Voici un exemple de requête GET qui envoie un message à une file d'attente Amazon SQS.

```
https://sqs.us-east-2.amazonaws.com/123456789012/MyQueue
?Action=SendMessage&MessageBody=Your%20message%20text
&Version=2012-11-05
&AUTHPARAMS
```

> **Note**
>
> Les noms et les URL des files d'attente sont sensibles à la casse.
> Dans la mesure où les requêtes GET sont des URL, vous devez coder en URL toutes les valeurs des paramètres. Les espaces ne sont pas autorisés dans les URL. Dès lors, chaque espace est encodé sous la forme `%20`. Pour faciliter la lecture, le reste de l'exemple n'a pas été encodé selon le format URL.

Envoi de requête POST

Une demande POST Amazon SQS envoie des paramètres de requête sous forme de formulaire dans le corps d'une demande HTTP.

L'exemple suivant illustre un en-tête HTTP avec `Content-Type` défini sur `application/x-www-form-urlencoded`.

```
POST /123456789012/MyQueue HTTP/1.1
```

```
Host: sqs.us-east-2.amazonaws.com
Content-Type: application/x-www-form-urlencoded
```

L'en-tête est suivi d'une requête GET `form-urlencoded` qui envoie un message à une file d'attente Amazon SQS. Chaque paramètre est séparé par une esperluette (&).

```
Action=SendMessage
&MessageBody=Your+Message+Text
&Expires=2020-10-15T12%3A00%3A00Z
&Version=2012-11-05
&AUTHPARAMS
```

> **Note**
>
> Seul l'en-tête HTTP `Content-Type` est obligatoire. L'élément *AUTHPARAMS* est le même que pour la requête GET.
> Votre client HTTP peut ajouter d'autres éléments à la requête HTTP, en fonction de la version HTTP du client.

Interprétation des réponses de l'API XML Amazon SQS

En réponse à une demande d'action, Amazon SQS renvoie une structure de données XML qui contient les résultats de la demande. Pour plus d'informations, consultez les actions individuelles dans la Référence d'API Amazon Simple Queue Service.

Rubriques

- Structure d'une réponse XML positive
- Structure d'une réponse d'erreur XML

Structure d'une réponse XML positive

Si la requête a abouti, l'élément de réponse principal porte le nom de l'action, mais avec `Response` ajouté (par exemple, *ActionName*`Response`).

Il contient les éléments enfants suivants :

- ***ActionName*`Result`** : contient un élément spécifique à l'action. Par exemple, l'élément `CreateQueueResult` contient l'élément `QueueUrl`, qui contient à son tour l'URL de la file d'attente créée.
- **`ResponseMetadata`** : contient le `RequestId` qui contient à son tour l'UUID (Universal Unique Identifier) de la requête.

Voici un exemple de réponse ayant abouti au format XML :

```
<CreateQueueResponse
   xmlns=https://sqs.us-east-2.amazonaws.com/doc/2012-11-05/
   xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
   xsi:type=CreateQueueResponse>
   <CreateQueueResult>
      <QueueUrl>https://sqs.us-east-2.amazonaws.com/770098461991/queue2</QueueUrl>
   </CreateQueueResult>
   <ResponseMetadata>
      <RequestId>cb919c0a-9bce-4afe-9b48-9bdf2412bb67</RequestId>
   </ResponseMetadata>
</CreateQueueResponse>
```

Structure d'une réponse d'erreur XML

Si une requête échoue, Amazon SQS renvoie toujours l'élément de réponse principal `ErrorResponse`. Il contient un élément `Error` et un élément `RequestId`.

L'élément `Error` contient les éléments enfants suivants :

- **`Type`** : indique si l'erreur est survenue au niveau du producteur ou du consommateur.
- **`Code`** : spécifie le type d'erreur.
- **`Message`** : spécifie la condition d'erreur dans un format lisible.
- **`Detail`** : (facultatif) spécifie des détails supplémentaires sur l'erreur.

L'élément `RequestId` contient l'UUID de la requête.

Voici un exemple de réponse d'erreur au format XML :

```
<ErrorResponse>
   <Error>
      <Type>Sender</Type>
```

```
      <Code>InvalidParameterValue</Code>
      <Message>
         Value (quename_nonalpha) for parameter QueueName is invalid.
         Must be an alphanumeric String of 1 to 80 in length.
      </Message>
   </Error>
   <RequestId>42d59b56-7407-4c4a-be0f-4c88daeea257</RequestId>
</ErrorResponse>
```

Authentification des demandes pour Amazon SQS

L'authentification est le processus d'identification et de vérification de la partie qui envoie une requête. Au cours de la première étape de l'authentification, AWS vérifie l'identité du créateur et si celui-ci est enregistré pour utiliser AWS (pour plus d'informations, consultez Étape 1 : créer un utilisateur Compte AWS et IAM). Ensuite, AWS respecte la procédure suivante :

1. Le producteur (expéditeur) obtient les autorisations requises.
2. Le créateur envoie une requête et l'autorisation au consommateur (destinataire).
3. Le consommateur utilise l'autorisation pour vérifier si le producteur a envoyé la requête.
4. L'une des situations suivantes se produit :
   - Si l'authentification réussit, le consommateur procède au traitement de la requête.
   - Si l'authentification échoue, le consommateur rejette la requête et renvoie une erreur.

Rubriques

Processus d'authentification de base avec HMAC-SHA

Lorsque vous accédez à Amazon SQS grâce à l'API Query, vous devez fournir les éléments suivants pour authentifier votre demande :

- L'identifiant de clé d'AWS accès qui vous identifie Compte AWS et qui est AWS utilisé pour rechercher votre clé d'accès secrète.

- Signature de demande HMAC-SHA, calculée à l'aide de votre clé d'accès secrète (code secret que seuls AWS et vous-même connaissez ; pour plus d'informations, consultez RFC2104). Le kit SDK AWS gère le processus de signature. Toutefois, si vous soumettez une requête de requête via HTTP ou HTTPS, vous devez inclure une signature dans chacune de ces requêtes.

  1. Dérivez une clé de signature de Signature Version 4. Pour plus d'informations, consultez Dérivation d'une clé de signature avec Java.

     > **Note**
     >
     > Amazon SQS prend en charge Signature version 4, qui offre une sécurité accrue basée sur SHA256 et de meilleures performances par rapport aux versions précédentes. Lorsque vous créez des applications qui utilisent Amazon SQS, optez pour Signature version 4.

  2. Codez en Base64 la signature de la requête. C'est ce que fait l'exemple de code Java suivant :

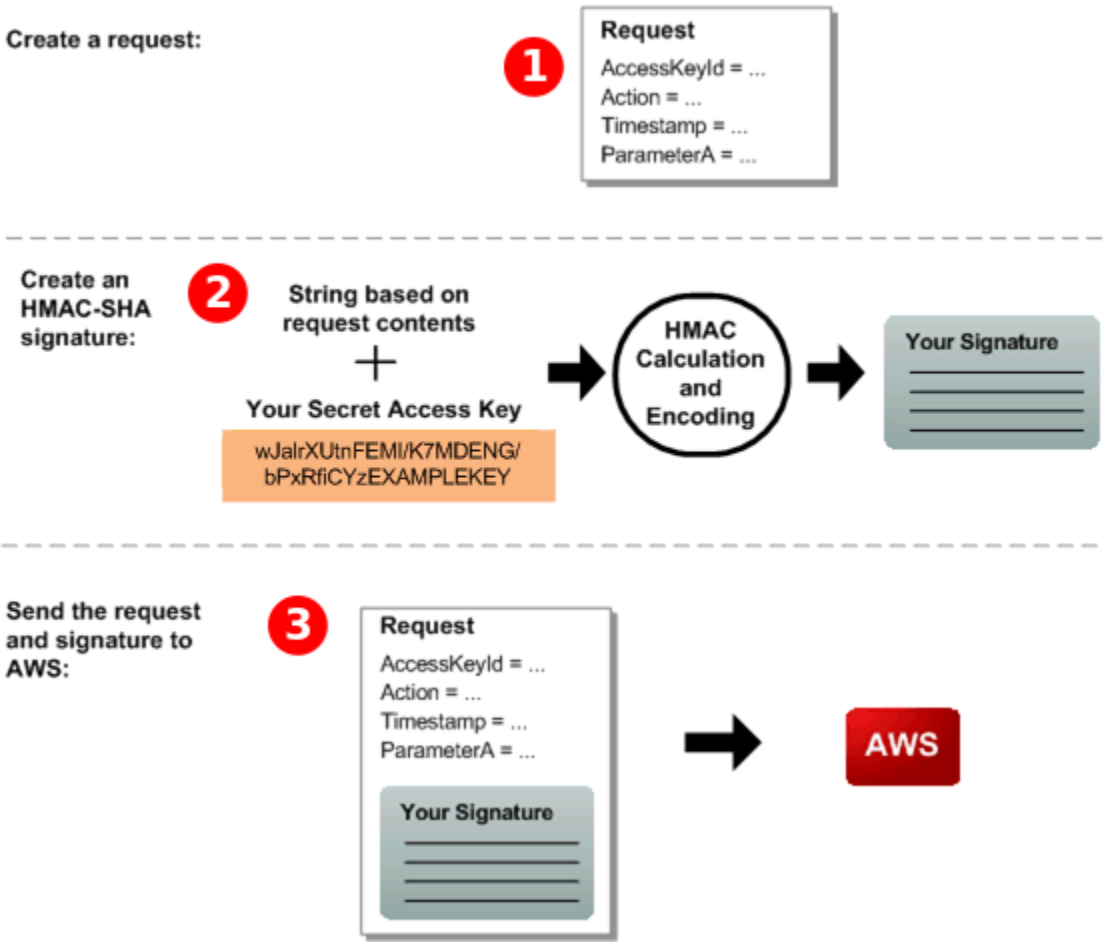
```
package amazon.webservices.common;

// Define common routines for encoding data in AWS requests.
public class Encoding {

    /* Perform base64 encoding of input bytes.
     * rawData is the array of bytes to be encoded.
     * return is the base64-encoded string representation of rawData.
     */
    public static String EncodeBase64(byte[] rawData) {
        return Base64.encodeBytes(rawData);
    }
}
```

- Horodatage (ou expiration) de la requête. L'horodatage que vous utilisez dans la requête doit être un objet `dateTime`, avec la date complète suivie des heures, des minutes et des secondes. Par exemple : `2007-01-31T23:59:59Z` Bien que ce ne soit pas obligatoire, nous vous recommandons de fournir l'heure système affichée dans la zone d'heure de l'heure universelle coordonnée (Heure de Greenwich).

> **Note**
>
> Assurez-vous que votre serveur affiche l'heure correcte. Si vous spécifiez un horodatage (plutôt qu'une expiration), la demande expire automatiquement 15 minutes après l'heure spécifiée (AWS ne traite pas les demandes dont l'horodatage est daté de plus de 15 minutes avant l'heure actuelle sur les serveurs). AWS
> Si vous utilisez .NET, vous ne devez pas envoyer des heures systèmes excessivement spécifiques, en raison des diverses propositions existant sur la façon de renoncer à une précision horaire absolue. Dans ce cas, vous devez créer manuellement les objets `dateTime` avec une précision ne dépassant pas une milliseconde.

Partie 1 : Demande de l'utilisateur

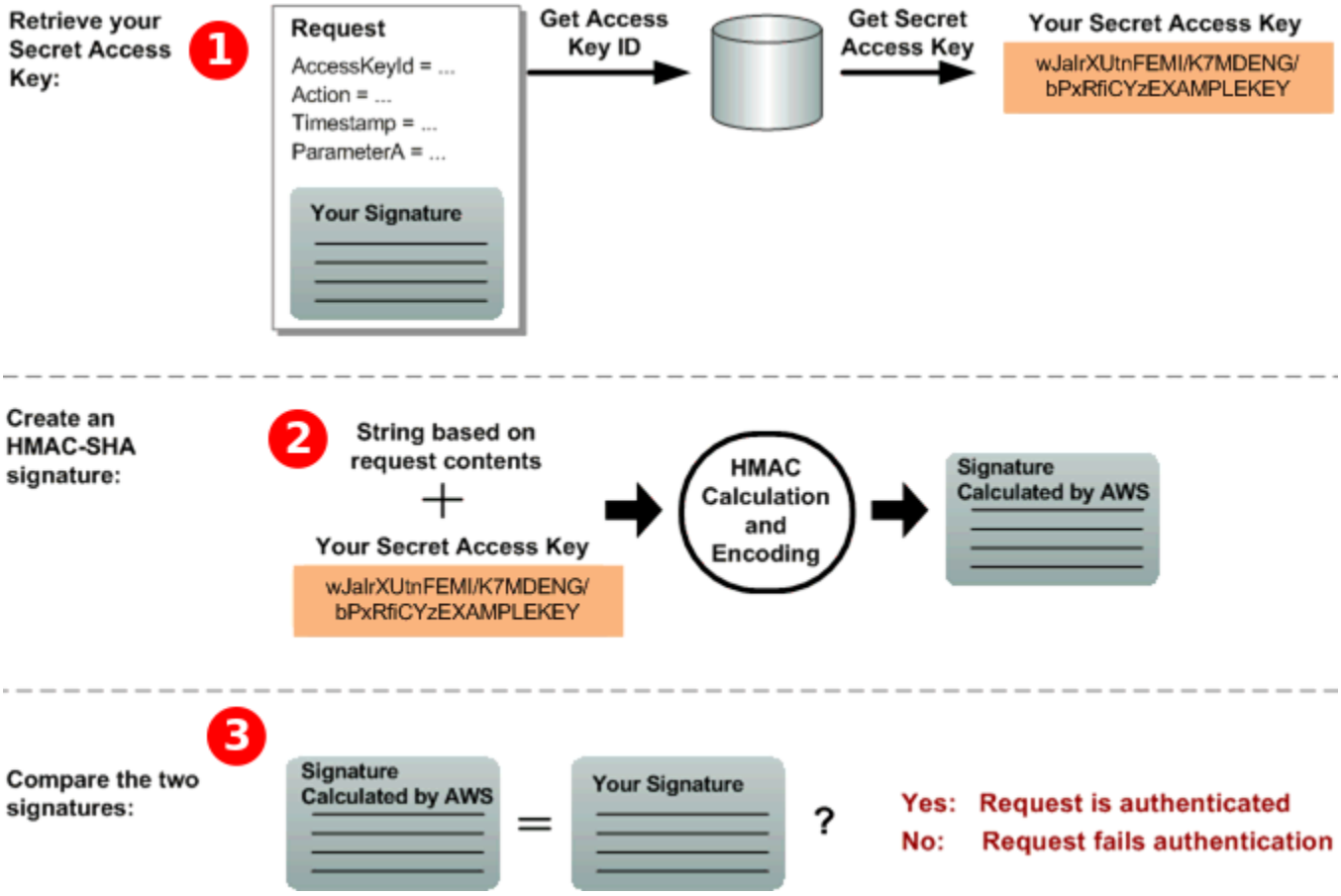
Voici le processus que vous devez suivre pour authentifier les AWS demandes à l'aide d'une signature de demande HMAC-SHA.



1. Créez une demande pour AWS.
2. Calculez une signature de code d'authentification du message avec hachage à clés (HMAC-SHA) à l'aide de votre clé d'accès secrète.
3. Incluez la signature et l'identifiant de votre clé d'accès dans la demande, puis envoyez la demande à AWS.

Partie 2 : La réponse de AWS

AWS lance le processus suivant en réponse.



1. AWS utilise l'ID de clé d'accès pour rechercher votre clé d'accès secrète.
2. AWS génère une signature à partir des données de la demande et de la clé d'accès secrète, en utilisant le même algorithme que celui que vous avez utilisé pour calculer la signature que vous avez envoyée dans la demande.
3. L'une des situations suivantes se produit :

- Si la signature AWS générée correspond à celle que vous avez envoyée dans la demande, AWS considérez que la demande est authentique.
- Si la comparaison échoue, la demande est rejetée et AWS renvoie une erreur.

Actions groupées Amazon SQS

Afin de réduire les coûts ou de manipuler jusqu'à 10 messages avec une seule action, vous pouvez utiliser les actions suivantes :

- `SendMessageBatch`
- `DeleteMessageBatch`
- `ChangeMessageVisibilityBatch`

Vous pouvez tirer parti de la fonctionnalité de traitement par lots à l'aide de l'API Query ou d'un AWS SDK prenant en charge les actions par lots Amazon SQS.

> **Note**
>
> La taille totale de tous les messages que vous envoyez en un seul `SendMessageBatch` appel ne peut pas dépasser 262 144 octets (256 KiB).
> Vous ne pouvez pas définir d'autorisations pour `SendMessageBatch`, `DeleteMessageBatch` ou `ChangeMessageVisibilityBatch` de manière explicite. Les autorisations définies pour `SendMessage`, `DeleteMessage`, or `ChangeMessageVisibility` s'appliquent également aux versions correspondantes de traitement par lots de ces actions.
> La console Amazon SQS ne prend pas en charge les actions groupées.

Rubriques

- Activation de la mise en mémoire tampon côté client et du traitement par lots des demandes avec Amazon SQS
- Augmenter le débit grâce à la mise à l'échelle horizontale et au traitement par lots d'actions avec Amazon SQS

Activation de la mise en mémoire tampon côté client et du traitement par lots des demandes avec Amazon SQS

Le kit AWS SDK for Java inclut `AmazonSQSBufferedAsyncClient` qui accède à Amazon SQS. Ce client facilite le traitement par lots des demandes à l'aide d'une mise en tampon côté client : les appels du client sont d'abord mis en tampon, puis transmis comme demande par lots à Amazon SQS.

La mise en tampon côté client autorise la mise en tampon et l'envoi en tant que demande par lots de 10 requêtes au maximum, ce qui réduit les coûts d'utilisation d'Amazon SQS et le nombre de requêtes envoyées. `AmazonSQSBufferedAsyncClient` met en tampon les appels synchrones et asynchrones. Les demandes par lots et la prise en charge de l'attente active de longue durée peuvent également permettre d'augmenter le débit. Pour plus d'informations, consultez Augmenter le débit grâce à la mise à l'échelle horizontale et au traitement par lots d'actions avec Amazon SQS.

Dans la mesure où `AmazonSQSBufferedAsyncClient` implémente la même interface qu'`AmazonSQSAsyncClient`, la migration d'`AmazonSQSAsyncClient` vers `AmazonSQSBufferedAsyncClient` requiert généralement très peu de changements du code existant.

> **Note**
>
> Le client asynchrone en mémoire tampon Amazon SQS ne prend actuellement pas en charge les files d'attente FIFO.

Rubriques

- Utilisation du client AmazonSQS BufferedAsync
- Configuration du client AmazonSQS BufferedAsync

Utilisation du client AmazonSQS BufferedAsync

Avant de commencer, complétez les étapes détaillées dans Configuration d'Amazon SQS.

> **Important**
>
> Le AWS SDK for Java 2.x n'est actuellement pas compatible avec le`AmazonSQSBufferedAsyncClient`.

Vous pouvez créer un nouveau `AmazonSQSBufferedAsyncClient` basé sur `AmazonSQSAsyncClient`, par exemple :

```
// Create the basic Amazon SQS async client
final AmazonSQSAsync sqsAsync = new AmazonSQSAsyncClient();

// Create the buffered client
final AmazonSQSAsync bufferedSqs = new AmazonSQSBufferedAsyncClient(sqsAsync);
```

Après avoir créé le nouveau `AmazonSQSBufferedAsyncClient`, utilisez-le pour envoyer plusieurs demandes à Amazon SQS (comme avec `AmazonSQSAsyncClient`), par exemple :

```
final CreateQueueRequest createRequest = new
 CreateQueueRequest().withQueueName("MyQueue");

final CreateQueueResult res = bufferedSqs.createQueue(createRequest);

final SendMessageRequest request = new SendMessageRequest();
final String body = "Your message text" + System.currentTimeMillis();
request.setMessageBody( body );
request.setQueueUrl(res.getQueueUrl());

final Future<SendMessageResult> sendResult = bufferedSqs.sendMessageAsync(request);

final ReceiveMessageRequest receiveRq = new ReceiveMessageRequest()
    .withMaxNumberOfMessages(1)
    .withQueueUrl(queueUrl);
final ReceiveMessageResult rx = bufferedSqs.receiveMessage(receiveRq);
```

Configuration du client AmazonSQS BufferedAsync

`AmazonSQSBufferedAsyncClient` est préconfiguré avec des paramètres qui fonctionnent dans la plupart des cas d'utilisation. Vous pouvez configurer davantage `AmazonSQSBufferedAsyncClient`, par exemple :

1. Créez une instance de la classe `QueueBufferConfig` avec les paramètres de configuration requis.
2. Fournissez l'instance au constructeur `AmazonSQSBufferedAsyncClient`.

```
// Create the basic Amazon SQS async client
```

```
final AmazonSQSAsync sqsAsync = new AmazonSQSAsyncClient();

final QueueBufferConfig config = new QueueBufferConfig()
    .withMaxInflightReceiveBatches(5)
    .withMaxDoneReceiveBatches(15);

// Create the buffered client
final AmazonSQSAsync bufferedSqs = new AmazonSQSBufferedAsyncClient(sqsAsync, config);
```

QueueBufferConfig paramètres de configuration

Paramètre	Valeur par défaut	Description
`longPoll`	`true`	Lorsque `longPoll` est défini sur `true`, `AmazonSQSBufferedAsyncClient` tente d'utiliser l'attente active de longue durée lors de la consommation des messages.
`longPollWaitTimeoutSeconds`	20 s	Durée maximale, en secondes, pendant laquelle un `ReceiveMessage` reste sur le serveur en attendant l'apparition de messages dans la file d'attente avant de renvoyer un résultat de réception vide. **Note** Ce paramètre n'a pas d'impact lorsque l'attente active de longue durée est désactivée.

Paramètre	Valeur par défaut	Description
`maxBatchOpenMs`	200 ms	Durée maximale, en millisecondes, pendant laquelle un appel sortant attend d'autres appels du même type pour le traitement par lots. Plus le paramètre est élevé, moins il faut de lots pour effectuer la même quantité de travail (toutefois, le premier appel d'un lot doit passer plus de temps à attendre). Lorsque ce paramètre est défini sur `0`, les requêtes envoyées n'attendent pas d'autres requêtes, ce qui a pour effet de désactiver le traitement par lots.

Paramètre	Valeur par défaut	Description
`maxBatchSize`	10 requêtes par lot	Nombre maximal de messages traités dans un même lot dans le cadre d'une seule requête. Plus ce paramètre est élevé, moins le nombre de lots requis pour effectuer le même nombre de requêtes est élevé. **Note** 10 demandes par lots est la valeur maximale autorisée pour Amazon SQS.
`maxBatchSizeBytes`	256 Kio	Taille maximale d'un lot de messages, en octets, que le client essaie d'envoyer à Amazon SQS. **Note** 256 KiB est la valeur maximale autorisée pour Amazon SQS.

Paramètre	Valeur par défaut	Description
`maxDoneReceiveBatches`	10 lots	Nombre maximal de lots de réception récupérés au préalable par `AmazonSQSBufferedAsyncClient` et stockés côté client. Plus le paramètre est élevé, plus il est possible de satisfaire un grand nombre de demandes sans devoir appeler Amazon SQS (toutefois, plus le nombre de messages récupérés au préalable est important, plus ils restent longtemps dans la mémoire tampon, ce qui entraîne l'expiration de leur délai de visibilité). **Note** `0`indique que la prélecture des messages est désactivée et que les messages sont consommés uniquement à la demande.

Paramètre	Valeur par défaut	Description
`maxInflightOutboundBatches`	5 lots	Nombre maximal de lots sortants actifs pouvant être traités en même temps. Plus ce paramètre est élevé, plus les lots sortants peuvent être envoyés rapidement (sous réserve d'autres quotas, tels que l'UC ou la bande passante) et plus le nombre de threads consommés par `AmazonSQSBufferedAsyncClient` est important.

Paramètre	Valeur par défaut	Description
`maxInflightReceiveBatches`	10 lots	Nombre maximum de lots de réception actifs pouvant être traités en même temps. Plus ce paramètre est élevé, plus le nombre de messages susceptibles d'être reçus est important (sous réserve d'autres quotas, tels que l'UC ou la bande passante) et plus le nombre de threads consommés par `AmazonSQSBufferedAsyncClient` est important. **Note** `0`indique que la prélecture des messages est désactivée et que les messages sont consommés uniquement à la demande.

Paramètre	Valeur par défaut	Description
`visibilityTimeoutSeconds`	-1	Lorsqu'une valeur positive autre que zéro a été définie pour ce paramètre, le délai de visibilité définit ici prévaut sur celui de la file d'attente à partir de laquelle les messages sont consommés. **Note** `-1` indique que le paramètre par défaut est sélectionné pour la file d'attente. Vous ne pouvez pas définir de délai de visibilité sur `0`.

Augmenter le débit grâce à la mise à l'échelle horizontale et au traitement par lots d'actions avec Amazon SQS

Les files d'attente Amazon SQS peuvent fournir un débit très élevé. Pour plus d'informations sur les quotas de débit, consultez Quotas de messages Amazon SQS.

Pour atteindre un débit élevé, vous devez effectuer une mise à l'échelle horizontale des producteurs et consommateurs de messages (ajouter des producteurs et des consommateurs supplémentaires).

Rubriques

- Mise à l'échelle horizontale
- Traitement par lots des actions
- Exemple d'utilisation de Java pour les requêtes en une seule opération et par lots

Mise à l'échelle horizontale

Vous accédez à Amazon SQS via un protocole de demande-réponse HTTP. Par conséquent, la latence de la demande (l'intervalle de temps entre la création d'une requête et la réception de la réponse) limite le débit que vous pouvez obtenir d'un seul thread avec une connexion unique. Par exemple, si la latence d'un client basé sur Amazon EC2 envoyant des demandes vers Amazon SQS dans la même région avoisine les 20 ms, le débit maximal d'un thread unique sur une seule connexion est en moyenne de 50 TPS.

La mise à l'échelle horizontale consiste à augmenter le nombre de producteurs de messages (émettant des requêtes `SendMessage`) et de consommateurs de messages (émettant des requêtes `ReceiveMessage` et `DeleteMessage`) afin d'augmenter le débit global de votre file d'attente. Vous pouvez mettre à l'échelle horizontalement de trois manières :

- Augmenter le nombre de threads par client
- Ajouter des clients
- Augmentez le nombre de threads par client et ajouter d'autres clients

Lorsque vous ajoutez des clients, vous créez essentiellement un gain de débit linéaire pour votre file d'attente. Ainsi, si vous doublez le nombre de clients, vous obtiendrez deux fois plus de débit.

> **Note**
>
> Lorsque vous procédez à une mise à l'échelle horizontale, assurez-vous que votre client Amazon SQS dispose de suffisamment de connexions ou de threads pour prendre en charge le nombre de producteurs et de consommateurs de messages qui envoient des requêtes et reçoivent des réponses simultanément. Par exemple, par défaut, les instances de la AWS SDK for Java `AmazonSQSClient` classe conservent au maximum 50 connexions à Amazon SQS. Pour créer des producteurs et consommateurs simultanés supplémentaires, vous devez modifier le nombre maximum de threads de producteurs et de consommateurs autorisés sur un objet `AmazonSQSClientBuilder`, par exemple :
>
> ```
> final AmazonSQS sqsClient = AmazonSQSClientBuilder.standard()
>         .withClientConfiguration(new ClientConfiguration()
>                 .withMaxConnections(producerCount + consumerCount))
>         .build();
> ```

Pour `AmazonSQSAsyncClient`, vous devez également veiller à ce que le nombre de threads disponibles soit suffisant.
Cet exemple ne fonctionne que pour Java v. 1.x.

Traitement par lots des actions

Le traitement par lots effectue davantage de travail au cours de chaque aller-retour vers le service (par exemple, lorsque vous envoyez plusieurs messages avec une seule requête `SendMessageBatch`). Les actions par lots Amazon SQS sont `SendMessageBatch`, `DeleteMessageBatch` et `ChangeMessageVisibilityBatch`. Pour profiter du traitement par lots sans modifier vos producteurs ou consommateurs, vous pouvez utiliser le Client asynchrone en mémoire tampon Amazon SQS.

Note

Dans la mesure où `ReceiveMessage` peut traiter 10 messages simultanément, il n'y a pas d'action `ReceiveMessageBatch`.

Le traitement par lots répartit la latence de l'action sur plusieurs messages dans une demande par lots, au lieu d'accepter la totalité de la latence pour un seul message (par exemple, une demande `SendMessage`). Dans la mesure où chaque échange avec le service inclut davantage de tâches à traiter, les requêtes par lots assurent une utilisation plus efficace des threads et connexions, ce qui améliore le débit.

Vous pouvez combiner la mise à l'échelle horizontale et le traitement par lots afin de proposer un débit avec moins de threads, de connexions et de requêtes que pour les requêtes de message individuelles. Vous pouvez utiliser des actions Amazon SQS par lots pour envoyer, recevoir ou supprimer jusqu'à 10 messages à la fois. Dans la mesure où Amazon SQS facture par demande, le traitement par lots peut vous aider à réduire considérablement vos coûts.

Le traitement par lots peut ajouter un peu de complexité pour votre application (par exemple, votre application doit accumuler les messages avant de les envoyer, ou doit parfois attendre plus longtemps pour une réponse). Il reste toutefois efficace dans les cas suivants :

- Votre application génère de nombreux messages en peu de temps, si bien que le délai d'attente n'est jamais très long.

- Un consommateur de messages récupère les messages dans une file d'attente à sa discrétion, contrairement à un système où des producteurs de messages classiques ont besoin d'envoyer des messages en réponse à des événements qu'ils ne contrôlent pas.

> **Important**
>
> Une requête de traitement par lots peut aboutir même si des messages individuels inclus dans cette requête n'ont pas pu être traités. Après une requête de traitement par lot, recherchez toujours d'éventuelles erreurs concernant des messages individuels et, le cas échéant, relancez l'action.

Exemple d'utilisation de Java pour les requêtes en une seule opération et par lots

Prérequis

Ajoutez les packages `aws-java-sdk-sqs.jar`, `aws-java-sdk-ec2.jar` et `commons-logging.jar` au chemin de classe de votre version Java. L'exemple suivant illustre ces dépendances dans un fichier `pom.xml` de projet Maven.

```
<dependencies>
    <dependency>
        <groupId>com.amazonaws</groupId>
        <artifactId>aws-java-sdk-sqs</artifactId>
        <version>LATEST</version>
    </dependency>
    <dependency>
        <groupId>com.amazonaws</groupId>
        <artifactId>aws-java-sdk-ec2</artifactId>
        <version>LATEST</version>
    </dependency>
    <dependency>
        <groupId>commons-logging</groupId>
        <artifactId>commons-logging</artifactId>
        <version>LATEST</version>
    </dependency>
</dependencies>
```

SimpleProducerConsumer.java

L'exemple de code Java suivant implémente un schéma simple consommateur-producteur. Le thread principal génère un certain nombre de threads producteurs et consommateurs qui traitent des messages de 1 Ko pendant une durée spécifiée. Cet exemple inclut des producteurs et des consommateurs qui émettent des requêtes d'opérations simples, et d'autres qui créent des requêtes de traitement par lot.

```
/*
 * Copyright 2010-2024 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

import com.amazonaws.AmazonClientException;
import com.amazonaws.ClientConfiguration;
import com.amazonaws.services.sqs.AmazonSQS;
import com.amazonaws.services.sqs.AmazonSQSClientBuilder;
import com.amazonaws.services.sqs.model.*;
import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;

import java.math.BigInteger;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import java.util.Scanner;
import java.util.concurrent.TimeUnit;
import java.util.concurrent.atomic.AtomicBoolean;
import java.util.concurrent.atomic.AtomicInteger;

/**
 * Start a specified number of producer and consumer threads, and produce-consume
```

```
 * for the least of the specified duration and 1 hour. Some messages can be left
 * in the queue because producers and consumers might not be in exact balance.
 */
public class SimpleProducerConsumer {

    // The maximum runtime of the program.
    private final static int MAX_RUNTIME_MINUTES = 60;
    private final static Log log = LogFactory.getLog(SimpleProducerConsumer.class);

    public static void main(String[] args) throws InterruptedException {

        final Scanner input = new Scanner(System.in);

        System.out.print("Enter the queue name: ");
        final String queueName = input.nextLine();

        System.out.print("Enter the number of producers: ");
        final int producerCount = input.nextInt();

        System.out.print("Enter the number of consumers: ");
        final int consumerCount = input.nextInt();

        System.out.print("Enter the number of messages per batch: ");
        final int batchSize = input.nextInt();

        System.out.print("Enter the message size in bytes: ");
        final int messageSizeByte = input.nextInt();

        System.out.print("Enter the run time in minutes: ");
        final int runTimeMinutes = input.nextInt();

        /*
         * Create a new instance of the builder with all defaults (credentials
         * and region) set automatically. For more information, see Creating
         * Service Clients in the AWS SDK for Java Developer Guide.
         */
        final ClientConfiguration clientConfiguration = new ClientConfiguration()
                .withMaxConnections(producerCount + consumerCount);

        final AmazonSQS sqsClient = AmazonSQSClientBuilder.standard()
                .withClientConfiguration(clientConfiguration)
                .build();

        final String queueUrl = sqsClient
```

```
                .getQueueUrl(new GetQueueUrlRequest(queueName)).getQueueUrl();

        // The flag used to stop producer, consumer, and monitor threads.
        final AtomicBoolean stop = new AtomicBoolean(false);

        // Start the producers.
        final AtomicInteger producedCount = new AtomicInteger();
        final Thread[] producers = new Thread[producerCount];
        for (int i = 0; i < producerCount; i++) {
            if (batchSize == 1) {
                producers[i] = new Producer(sqsClient, queueUrl, messageSizeByte,
                        producedCount, stop);
            } else {
                producers[i] = new BatchProducer(sqsClient, queueUrl, batchSize,
                        messageSizeByte, producedCount,
                        stop);
            }
            producers[i].start();
        }

        // Start the consumers.
        final AtomicInteger consumedCount = new AtomicInteger();
        final Thread[] consumers = new Thread[consumerCount];
        for (int i = 0; i < consumerCount; i++) {
            if (batchSize == 1) {
                consumers[i] = new Consumer(sqsClient, queueUrl, consumedCount,
                        stop);
            } else {
                consumers[i] = new BatchConsumer(sqsClient, queueUrl, batchSize,
                        consumedCount, stop);
            }
            consumers[i].start();
        }

        // Start the monitor thread.
        final Thread monitor = new Monitor(producedCount, consumedCount, stop);
        monitor.start();

        // Wait for the specified amount of time then stop.
        Thread.sleep(TimeUnit.MINUTES.toMillis(Math.min(runTimeMinutes,
                MAX_RUNTIME_MINUTES)));
        stop.set(true);

        // Join all threads.
```

```
        for (int i = 0; i < producerCount; i++) {
            producers[i].join();
        }

        for (int i = 0; i < consumerCount; i++) {
            consumers[i].join();
        }

        monitor.interrupt();
        monitor.join();
    }

    private static String makeRandomString(int sizeByte) {
        final byte[] bs = new byte[(int) Math.ceil(sizeByte * 5 / 8)];
        new Random().nextBytes(bs);
        bs[0] = (byte) ((bs[0] | 64) & 127);
        return new BigInteger(bs).toString(32);
    }

    /**
     * The producer thread uses {@code SendMessage}
     * to send messages until it is stopped.
     */
    private static class Producer extends Thread {
        final AmazonSQS sqsClient;
        final String queueUrl;
        final AtomicInteger producedCount;
        final AtomicBoolean stop;
        final String theMessage;

        Producer(AmazonSQS sqsQueueBuffer, String queueUrl, int messageSizeByte,
                 AtomicInteger producedCount, AtomicBoolean stop) {
            this.sqsClient = sqsQueueBuffer;
            this.queueUrl = queueUrl;
            this.producedCount = producedCount;
            this.stop = stop;
            this.theMessage = makeRandomString(messageSizeByte);
        }

        /*
         * The producedCount object tracks the number of messages produced by
         * all producer threads. If there is an error, the program exits the
         * run() method.
         */
```

```
        public void run() {
            try {
                while (!stop.get()) {
                    sqsClient.sendMessage(new SendMessageRequest(queueUrl,
                            theMessage));
                    producedCount.incrementAndGet();
                }
            } catch (AmazonClientException e) {
                /*
                 * By default, AmazonSQSClient retries calls 3 times before
                 * failing. If this unlikely condition occurs, stop.
                 */
                log.error("Producer: " + e.getMessage());
                System.exit(1);
            }
        }
    }

    /**
     * The producer thread uses {@code SendMessageBatch}
     * to send messages until it is stopped.
     */
    private static class BatchProducer extends Thread {
        final AmazonSQS sqsClient;
        final String queueUrl;
        final int batchSize;
        final AtomicInteger producedCount;
        final AtomicBoolean stop;
        final String theMessage;

        BatchProducer(AmazonSQS sqsQueueBuffer, String queueUrl, int batchSize,
                      int messageSizeByte, AtomicInteger producedCount,
                      AtomicBoolean stop) {
            this.sqsClient = sqsQueueBuffer;
            this.queueUrl = queueUrl;
            this.batchSize = batchSize;
            this.producedCount = producedCount;
            this.stop = stop;
            this.theMessage = makeRandomString(messageSizeByte);
        }

        public void run() {
            try {
                while (!stop.get()) {
```

```
                final SendMessageBatchRequest batchRequest =
                        new SendMessageBatchRequest().withQueueUrl(queueUrl);

                final List<SendMessageBatchRequestEntry> entries =
                        new ArrayList<SendMessageBatchRequestEntry>();
                for (int i = 0; i < batchSize; i++)
                    entries.add(new SendMessageBatchRequestEntry()
                            .withId(Integer.toString(i))
                            .withMessageBody(theMessage));
                batchRequest.setEntries(entries);

                final SendMessageBatchResult batchResult =
                        sqsClient.sendMessageBatch(batchRequest);
                producedCount.addAndGet(batchResult.getSuccessful().size());

                /*
                 * Because SendMessageBatch can return successfully, but
                 * individual batch items fail, retry the failed batch items.
                 */
                if (!batchResult.getFailed().isEmpty()) {
                    log.warn("Producer: retrying sending "
                            + batchResult.getFailed().size() + " messages");
                    for (int i = 0, n = batchResult.getFailed().size();
                         i < n; i++) {
                        sqsClient.sendMessage(new
                                SendMessageRequest(queueUrl, theMessage));
                        producedCount.incrementAndGet();
                    }
                }
            }
        } catch (AmazonClientException e) {
            /*
             * By default, AmazonSQSClient retries calls 3 times before
             * failing. If this unlikely condition occurs, stop.
             */
            log.error("BatchProducer: " + e.getMessage());
            System.exit(1);
        }
    }
}

/**
 * The consumer thread uses {@code ReceiveMessage} and {@code DeleteMessage}
 * to consume messages until it is stopped.
```

```
     */
    private static class Consumer extends Thread {
        final AmazonSQS sqsClient;
        final String queueUrl;
        final AtomicInteger consumedCount;
        final AtomicBoolean stop;

        Consumer(AmazonSQS sqsClient, String queueUrl, AtomicInteger consumedCount,
                 AtomicBoolean stop) {
            this.sqsClient = sqsClient;
            this.queueUrl = queueUrl;
            this.consumedCount = consumedCount;
            this.stop = stop;
        }

        /*
         * Each consumer thread receives and deletes messages until the main
         * thread stops the consumer thread. The consumedCount object tracks the
         * number of messages that are consumed by all consumer threads, and the
         * count is logged periodically.
         */
        public void run() {
            try {
                while (!stop.get()) {
                    try {
                        final ReceiveMessageResult result = sqsClient
                                .receiveMessage(new
                                        ReceiveMessageRequest(queueUrl));

                        if (!result.getMessages().isEmpty()) {
                            final Message m = result.getMessages().get(0);
                            sqsClient.deleteMessage(new
                                    DeleteMessageRequest(queueUrl,
                                    m.getReceiptHandle()));
                            consumedCount.incrementAndGet();
                        }
                    } catch (AmazonClientException e) {
                        log.error(e.getMessage());
                    }
                }
            } catch (AmazonClientException e) {
                /*
                 * By default, AmazonSQSClient retries calls 3 times before
                 * failing. If this unlikely condition occurs, stop.
```

```
                 */
                log.error("Consumer: " + e.getMessage());
                System.exit(1);
            }
        }
    }

    /**
     * The consumer thread uses {@code ReceiveMessage} and {@code
     * DeleteMessageBatch} to consume messages until it is stopped.
     */
    private static class BatchConsumer extends Thread {
        final AmazonSQS sqsClient;
        final String queueUrl;
        final int batchSize;
        final AtomicInteger consumedCount;
        final AtomicBoolean stop;

        BatchConsumer(AmazonSQS sqsClient, String queueUrl, int batchSize,
                      AtomicInteger consumedCount, AtomicBoolean stop) {
            this.sqsClient = sqsClient;
            this.queueUrl = queueUrl;
            this.batchSize = batchSize;
            this.consumedCount = consumedCount;
            this.stop = stop;
        }

        public void run() {
            try {
                while (!stop.get()) {
                    final ReceiveMessageResult result = sqsClient
                            .receiveMessage(new ReceiveMessageRequest(queueUrl)
                                    .withMaxNumberOfMessages(batchSize));

                    if (!result.getMessages().isEmpty()) {
                        final List<Message> messages = result.getMessages();
                        final DeleteMessageBatchRequest batchRequest =
                                new DeleteMessageBatchRequest()
                                        .withQueueUrl(queueUrl);

                        final List<DeleteMessageBatchRequestEntry> entries =
                                new ArrayList<DeleteMessageBatchRequestEntry>();
                        for (int i = 0, n = messages.size(); i < n; i++)
                            entries.add(new DeleteMessageBatchRequestEntry()
```

```
                                .withId(Integer.toString(i))
                                .withReceiptHandle(messages.get(i)
                                        .getReceiptHandle()));
                        batchRequest.setEntries(entries);

                        final DeleteMessageBatchResult batchResult = sqsClient
                                .deleteMessageBatch(batchRequest);
                        consumedCount.addAndGet(batchResult.getSuccessful().size());

                        /*
                         * Because DeleteMessageBatch can return successfully,
                         * but individual batch items fail, retry the failed
                         * batch items.
                         */
                        if (!batchResult.getFailed().isEmpty()) {
                            final int n = batchResult.getFailed().size();
                            log.warn("Producer: retrying deleting " + n
                                    + " messages");
                            for (BatchResultErrorEntry e : batchResult
                                    .getFailed()) {

                                sqsClient.deleteMessage(
                                        new DeleteMessageRequest(queueUrl,
                                                messages.get(Integer
                                                        .parseInt(e.getId()))
                                                        .getReceiptHandle()));

                                consumedCount.incrementAndGet();
                            }
                        }
                    }
                } catch (AmazonClientException e) {
                    /*
                     * By default, AmazonSQSClient retries calls 3 times before
                     * failing. If this unlikely condition occurs, stop.
                     */
                    log.error("BatchConsumer: " + e.getMessage());
                    System.exit(1);
                }
            }
        }

        /**
```

```
     * This thread prints every second the number of messages produced and
     * consumed so far.
     */
    private static class Monitor extends Thread {
        private final AtomicInteger producedCount;
        private final AtomicInteger consumedCount;
        private final AtomicBoolean stop;

        Monitor(AtomicInteger producedCount, AtomicInteger consumedCount,
                AtomicBoolean stop) {
            this.producedCount = producedCount;
            this.consumedCount = consumedCount;
            this.stop = stop;
        }

        public void run() {
            try {
                while (!stop.get()) {
                    Thread.sleep(1000);
                    log.info("produced messages = " + producedCount.get()
                            + ", consumed messages = " + consumedCount.get());
                }
            } catch (InterruptedException e) {
                // Allow the thread to exit.
            }
        }
    }
}
```

Surveillance des métriques de volume pour l'exemple exécuté

Amazon SQS génère automatiquement des métriques de volume pour les messages envoyés, reçus et supprimés. Vous pouvez accéder à ces statistiques et à d'autres via l'onglet Surveillance de votre file d'attente ou sur la CloudWatch console.

> **Note**
>
> Après le démarrage de la file d'attente, il faut patienter jusqu'à 15 minutes pour que ces métriques soient disponibles.

Utilisation de JMS et d'Amazon SQS

La bibliothèque de messagerie Java Amazon SQS est une interface de service de message Java (JMS) pour Amazon SQS qui vous permet de tirer parti d'Amazon SQS dans les applications qui utilisent déjà JMS. L'interface vous permet d'utiliser Amazon SQS en tant que fournisseur JMS avec peu de changements de code. Avec le kit AWS SDK for Java, la bibliothèque de messagerie Java Amazon SQS vous permet de créer des connexions et des sessions JMS, ainsi que des producteurs et des consommateurs qui envoient et reçoivent des messages de files d'attente Amazon SQS.

La bibliothèque prend en charge l'envoi et la réception de messages vers une file d'attente (le point-to-point modèle JMS) conformément à la spécification JMS 1.1. Elle prend également en charge l'envoi de messages de texte, d'octets ou d'objets de façon synchrone aux files d'attente Amazon SQS. ainsi que la réception d'objets de façon synchrone ou asynchrone.

Pour plus d'informations sur les fonctionnalités de la bibliothèque de messagerie Java Amazon SQS conformes à la spécification JMS 1.1, consultez Implémentations de JMS 1.1 prises en charge par Amazon SQS et les FAQ d'Amazon SQS.

Rubriques

- Conditions requises pour travailler avec JMS et Amazon SQS
- Premiers pas avec la bibliothèque de messagerie Java Amazon SQS
- Utilisation du service de message Java avec d'autres clients Amazon SQS
- Exemples Java fonctionnels pour l'utilisation de JMS avec les files d'attente standard Amazon SQS
- Implémentations de JMS 1.1 prises en charge par Amazon SQS

Conditions requises pour travailler avec JMS et Amazon SQS

Avant de commencer, les prérequis suivants doivent être remplis :

- SDK pour Java

  Il existe deux façons d'inclure le kit SDK pour Java dans votre projet :

  - Téléchargez et installez le kit SDK pour Java.
  - Utilisez Maven pour obtenir la bibliothèque de messagerie Java Amazon SQS.

> **Note**
>
> Le kit SDK pour Java est fourni en tant que dépendance.
> Le kit SDK pour Java et la bibliothèque client étendue Amazon SQS pour Java nécessitent le kit de développement J2SE 8.0 ou version ultérieure.

Pour plus d'informations sur le téléchargement du kit SDK pour Java, consultez SDK pour Java.

- Bibliothèque de messagerie Java Amazon SQS

  Si vous n'utilisez pas Maven, vous devez ajouter le package `amazon-sqs-java-messaging-lib.jar` au chemin de classe Java. Pour plus d'informations sur le téléchargement de la bibliothèque, consultez Bibliothèque de messagerie Java Amazon SQS.

  > **Note**
  >
  > La bibliothèque de messagerie Java Amazon SQS inclut la prise en charge de Maven et du framework Spring.
  > Pour obtenir des exemples de code qui utilisent Maven, le framework Spring et la bibliothèque de messagerie Java Amazon SQS, consultez la section Exemples Java fonctionnels pour l'utilisation de JMS avec les files d'attente standard Amazon SQS.
  >
  > ```
  > <dependency>
  >   <groupId>com.amazonaws</groupId>
  >   <artifactId>amazon-sqs-java-messaging-lib</artifactId>
  >   <version>1.0.4</version>
  >   <type>jar</type>
  > </dependency>
  > ```

- File d'attente Amazon SQS

  Créez une file d'attente à l'aide de l'`CreateQueue`API AWS Management Console pour Amazon SQS, ou du client Amazon SQS encapsulé inclus dans la bibliothèque de messagerie Java Amazon SQS.

  - Pour plus d'informations sur la création d'une file d'attente avec Amazon SQS à l'aide de la AWS Management Console ou de l'API `CreateQueue`, consultez Création d'une file d'attente.

- Pour plus d'informations sur l'utilisation de la bibliothèque de messagerie Java Amazon SQS, consultez Premiers pas avec la bibliothèque de messagerie Java Amazon SQS.

Premiers pas avec la bibliothèque de messagerie Java Amazon SQS

Pour commencer à utiliser le service de messagerie Java (JMS) avec Amazon SQS, utilisez les exemples de code de cette section. Les sections suivantes montrent comment créer une connexion et une session JMS, et comment envoyer et recevoir un message.

L'objet client Amazon SQS encapsulé inclus dans la bibliothèque de messagerie Java Amazon SQS vérifie l'existence d'une file d'attente Amazon SQS. Si elle n'existe pas, le client la crée.

Création d'une connexion JMS

1. Créez une fabrique de connexions et appelez la méthode `createConnection` par rapport à cette fabrique.

```
// Create a new connection factory with all defaults (credentials and region) set
 automatically
SQSConnectionFactory connectionFactory = new SQSConnectionFactory(
        new ProviderConfiguration(),
        AmazonSQSClientBuilder.defaultClient()
        );

// Create the connection.
SQSConnection connection = connectionFactory.createConnection();
```

   La classe `SQSConnection` étend `javax.jms.Connection`. Avec les méthodes de connexion standard JMS, `SQSConnection` propose des méthodes supplémentaires, comme `getAmazonSQSClient` et `getWrappedAmazonSQSClient`. Ces deux méthodes vous permettent d'effectuer des opérations administratives non comprises dans la spécification JMS, telles que la création de files d'attente. Toutefois, la méthode `getWrappedAmazonSQSClient` fournit également une version encapsulée du client Amazon SQS utilisé par la connexion en cours. Le wrapper convertit toutes les exceptions en `JMSException` depuis le client, afin d'en faciliter l'utilisation par le code existant qui attend des occurrences `JMSException`.

2. Vous pouvez utiliser les objets client renvoyés par `getAmazonSQSClient` et `getWrappedAmazonSQSClient` pour effectuer des opérations administratives non comprises dans la spécification JMS (par exemple, vous pouvez créer une file d'attente Amazon SQS).

   Si votre code existant attend des exceptions JMS, vous devez alors utiliser `getWrappedAmazonSQSClient` :

   - Si vous utilisez `getWrappedAmazonSQSClient`, l'objet client renvoyé transforme toutes les exceptions en exceptions JMS.
   - Si vous utilisez `getAmazonSQSClient`, les exceptions sont toutes des exceptions Amazon SQS.

Création d'une file d'attente Amazon SQS

L'objet client encapsulé vérifie s'il existe une file d'attente Amazon SQS.

Si elle n'existe pas, le client la crée. Si la file d'attente existe, la fonction ne renvoie rien. Pour plus d'informations, consultez la section « Création de la file d'attente selon les besoins » section dans l'exemple TextMessageSender.java.

Pour créer une file d'attente standard

```
// Get the wrapped client
AmazonSQSMessagingClientWrapper client = connection.getWrappedAmazonSQSClient();

// Create an SQS queue named MyQueue, if it doesn't already exist
if (!client.queueExists("MyQueue")) {
    client.createQueue("MyQueue");
}
```

Pour créer une file d'attente FIFO

```
// Get the wrapped client
AmazonSQSMessagingClientWrapper client = connection.getWrappedAmazonSQSClient();

// Create an Amazon SQS FIFO queue named MyQueue.fifo, if it doesn't already exist
if (!client.queueExists("MyQueue.fifo")) {
    Map<String, String> attributes = new HashMap<String, String>();
    attributes.put("FifoQueue", "true");
```

```
    attributes.put("ContentBasedDeduplication", "true");
    client.createQueue(new
 CreateQueueRequest().withQueueName("MyQueue.fifo").withAttributes(attributes));
}
```

> **Note**
>
> Le nom d'une file d'attente FIFO doit se terminer par le suffixe `.fifo`.
> Pour plus d'informations sur l'attribut `ContentBasedDeduplication`, consultez Traitement effectué en une seule fois dans Amazon SQS.

Envoi de messages de façon synchrone

1. Lorsque la connexion et la file d'attente Amazon SQS sous-jacente sont prêtes, créez une session JMS non traitée avec le mode `AUTO_ACKNOWLEDGE`.

   ```
   // Create the nontransacted session with AUTO_ACKNOWLEDGE mode
   Session session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);
   ```

2. Pour envoyer un SMS à la file d'attente, créez une identité de file d'attente JMS et un producteur de message.

   ```
   // Create a queue identity and specify the queue name to the session
   Queue queue = session.createQueue("MyQueue");

   // Create a producer for the 'MyQueue'
   MessageProducer producer = session.createProducer(queue);
   ```

3. Créez un SMS et envoyez-le à la file d'attente.

   - Pour envoyer un message à une file d'attente standard, vous n'avez pas besoin de définir de paramètres supplémentaires.

     ```
     // Create the text message
     TextMessage message = session.createTextMessage("Hello World!");

     // Send the message
     producer.send(message);
     System.out.println("JMS Message " + message.getJMSMessageID());
     ```

- Pour envoyer un message à une file d'attente FIFO, vous devez définir l'ID de groupe de messages. Vous pouvez aussi définir un ID de déduplication de message. Pour plus d'informations, consultez Termes clés d'Amazon SQS.

```
// Create the text message
TextMessage message = session.createTextMessage("Hello World!");

// Set the message group ID
message.setStringProperty("JMSXGroupID", "Default");

// You can also set a custom message deduplication ID
// message.setStringProperty("JMS_SQS_DeduplicationId", "hello");
// Here, it's not needed because content-based deduplication is enabled for the
 queue

// Send the message
producer.send(message);
System.out.println("JMS Message " + message.getJMSMessageID());
System.out.println("JMS Message Sequence Number " +
 message.getStringProperty("JMS_SQS_SequenceNumber"));
```

Réception des messages de façon synchrone

1. Pour recevoir les messages, créez un consommateur correspondant à la même file d'attente et appelez la méthode `start`.

   Vous pouvez appeler à tout moment la méthode `start` avec cette connexion. Toutefois, le consommateur ne reçoit aucun message tant que vous ne l'appelez pas.

```
// Create a consumer for the 'MyQueue'
MessageConsumer consumer = session.createConsumer(queue);
// Start receiving incoming messages
connection.start();
```

2. Appelez la méthode `receive` au niveau du consommateur avec un délai d'attente d'une seconde, puis imprimez le contenu du message reçu.

   - Après avoir reçu un message d'une file d'attente standard, vous pouvez accéder à son contenu.

```
// Receive a message from 'MyQueue' and wait up to 1 second
Message receivedMessage = consumer.receive(1000);

// Cast the received message as TextMessage and display the text
if (receivedMessage != null) {
    System.out.println("Received: " + ((TextMessage) receivedMessage).getText());
}
```

- Après avoir reçu un message d'une file d'attente FIFO, vous pouvez accéder à son contenu et à d'autres attributs de message spécifiques à FIFO, comme l'ID de groupe de messages, l'ID de déduplication du message et le numéro de séquence. Pour plus d'informations, consultez Termes clés d'Amazon SQS.

```
// Receive a message from 'MyQueue' and wait up to 1 second
Message receivedMessage = consumer.receive(1000);

// Cast the received message as TextMessage and display the text
if (receivedMessage != null) {
    System.out.println("Received: " + ((TextMessage) receivedMessage).getText());
    System.out.println("Group id: " +
 receivedMessage.getStringProperty("JMSXGroupID"));
    System.out.println("Message deduplication id: " +
 receivedMessage.getStringProperty("JMS_SQS_DeduplicationId"));
    System.out.println("Message sequence number: " +
 receivedMessage.getStringProperty("JMS_SQS_SequenceNumber"));
}
```

3. Fermez la connexion et la session.

```
// Close the connection (and the session).
connection.close();
```

La sortie ressemble à ce qui suit:

```
JMS Message ID:8example-588b-44e5-bbcf-d816example2
Received: Hello World!
```

> **Note**
>
> Vous pouvez utiliser le framework Spring pour initialiser ces objets.
> Pour plus d'informations, consultez `SpringExampleConfiguration.xml`, `SpringExample.java` et les autres classes d'assistance sous `ExampleConfiguration.java` et `ExampleCommon.java` dans la section Exemples Java fonctionnels pour l'utilisation de JMS avec les files d'attente standard Amazon SQS.

Pour obtenir des exemples complets d'envoi et de réception d'objets, consultez les sections TextMessageSender.java et SyncMessageReceiver.java.

Réception des messages de façon asynchrone

Dans l'exemple de la section Premiers pas avec la bibliothèque de messagerie Java Amazon SQS, un message est envoyé à `MyQueue` et reçu de façon synchrone.

L'exemple suivant montre comment recevoir les messages de façon asynchrone via un écouteur.

1. Implémentez l'interface `MessageListener`.

```
class MyListener implements MessageListener {

    @Override
    public void onMessage(Message message) {
        try {
            // Cast the received message as TextMessage and print the text to
 screen.
            System.out.println("Received: " + ((TextMessage) message).getText());
        } catch (JMSException e) {
            e.printStackTrace();
        }
    }
}
```

   La méthode onMessage de l'interface `MessageListener` est appelée lorsque vous recevez un message. Dans cette implémentation d'écouteur, le texte stocké dans le message est imprimé.

2. Au lieu d'appeler explicitement la méthode `receive` au niveau du consommateur, sélectionnez une instance de l'implémentation `MyListener` pour l'écouteur de messages du consommateur. Le thread principal attend pendant une seconde.

```
// Create a consumer for the 'MyQueue'.
MessageConsumer consumer = session.createConsumer(queue);

// Instantiate and set the message listener for the consumer.
consumer.setMessageListener(new MyListener());

// Start receiving incoming messages.
connection.start();

// Wait for 1 second. The listener onMessage() method is invoked when a message is
 received.
Thread.sleep(1000);
```

Les autres étapes sont identiques à celles de l'exemple Premiers pas avec la bibliothèque de messagerie Java Amazon SQS. Pour obtenir un exemple complet de consommateur asynchrone, consultez `AsyncMessageReceiver.java` dans Exemples Java fonctionnels pour l'utilisation de JMS avec les files d'attente standard Amazon SQS.

La sortie correspondant à cet exemple est similaire à l'exemple suivant :

```
JMS Message ID:8example-588b-44e5-bbcf-d816example2
Received: Hello World!
```

Utilisation du mode de reconnaissance du client

L'exemple de la section Premiers pas avec la bibliothèque de messagerie Java Amazon SQS utilise le mode `AUTO_ACKNOWLEDGE` où chaque message reçu est accepté automatiquement (et, par conséquent, supprimé de la file d'attente Amazon SQS sous-jacente).

1. Pour confirmer explicitement les messages après leur traitement, vous devez créer la session en mode `CLIENT_ACKNOWLEDGE`.

   ```
   // Create the non-transacted session with CLIENT_ACKNOWLEDGE mode.
   Session session = connection.createSession(false, Session.CLIENT_ACKNOWLEDGE);
   ```

2. Lorsque le message est reçu, affichez-le, puis reconnaissez-le explicitement.

   ```
   // Cast the received message as TextMessage and print the text to screen. Also
    acknowledge the message.
   ```

```
if (receivedMessage != null) {
    System.out.println("Received: " + ((TextMessage) receivedMessage).getText());
    receivedMessage.acknowledge();
    System.out.println("Acknowledged: " + message.getJMSMessageID());
}
```

> **Note**
>
> Avec ce mode, lorsqu'un message est reconnu, tous les messages reçus avant celui-ci sont implicitement reconnus aussi. Par exemple, si 10 messages sont reçus et que le 10e message est reconnu (dans l'ordre dans lequel les messages sont reçus), les 9 messages précédents sont également reconnus.

Les autres étapes sont identiques à celles de l'exemple Premiers pas avec la bibliothèque de messagerie Java Amazon SQS. Pour obtenir un exemple complet de consommateur synchrone avec le mode de reconnaissance du client, consultez `SyncMessageReceiverClientAcknowledge.java` dans Exemples Java fonctionnels pour l'utilisation de JMS avec les files d'attente standard Amazon SQS.

La sortie correspondant à cet exemple est similaire à l'exemple suivant :

```
JMS Message ID:4example-aa0e-403f-b6df-5e02example5
Received: Hello World!
Acknowledged: ID:4example-aa0e-403f-b6df-5e02example5
```

Utilisation du mode de reconnaissance indépendamment de l'ordre de réception

Lorsque vous utilisez le mode `CLIENT_ACKNOWLEDGE`, tous les messages reçus avant un message explicitement reconnu sont reconnus automatiquement. Pour plus d'informations, consultez Utilisation du mode de reconnaissance du client.

La bibliothèque de messagerie Java Amazon SQS propose un autre mode de reconnaissance. Lorsque vous utilisez le mode `UNORDERED_ACKNOWLEDGE`, tous les messages reçus doivent être individuellement et explicitement reconnus par le client, quel que soit leur ordre de réception. Pour ce faire, créez une session avec le mode `UNORDERED_ACKNOWLEDGE`.

```
// Create the non-transacted session with UNORDERED_ACKNOWLEDGE mode.
```

```
Session session = connection.createSession(false, SQSSession.UNORDERED_ACKNOWLEDGE);
```

Les étapes suivantes sont identiques à celles de l'exemple Utilisation du mode de reconnaissance du client. Pour obtenir un exemple complet de consommateur synchrone avec le mode UNORDERED_ACKNOWLEDGE, consultez SyncMessageReceiverUnorderedAcknowledge.java.

Dans cet exemple, la sortie est similaire à l'exemple suivant :

```
JMS Message ID:dexample-73ad-4adb-bc6c-4357example7
Received: Hello World!
Acknowledged: ID:dexample-73ad-4adb-bc6c-4357example7
```

Utilisation du service de message Java avec d'autres clients Amazon SQS

L'utilisation du client Amazon SQS Java Message Service (JMS) avec le AWS SDK limite la taille des messages Amazon SQS à 256 Ko. Cependant, vous pouvez créer un fournisseur JMS avec n'importe quel client Amazon SQS. Par exemple, vous pouvez utiliser le client JMS avec la bibliothèque client étendue Amazon SQS pour Java afin d'envoyer un message Amazon SQS qui contient une référence à une charge utile de message (jusqu'à 2 Go) dans Amazon S3. Pour plus d'informations, consultez Gestion de messages Amazon SQS volumineux à l'aide de Java et Amazon S3.

L'exemple de code Java suivant crée le fournisseur JMS pour la bibliothèque client étendue :

```
AmazonS3 s3 = new AmazonS3Client(credentials);
Region s3Region = Region.getRegion(Regions.US_WEST_2);
s3.setRegion(s3Region);

// Set the Amazon S3 bucket name, and set a lifecycle rule on the bucket to
// permanently delete objects a certain number of days after each object's creation
 date.
// Next, create the bucket, and enable message objects to be stored in the bucket.
BucketLifecycleConfiguration.Rule expirationRule = new
 BucketLifecycleConfiguration.Rule();
expirationRule.withExpirationInDays(14).withStatus("Enabled");
BucketLifecycleConfiguration lifecycleConfig = new
 BucketLifecycleConfiguration().withRules(expirationRule);

s3.createBucket(s3BucketName);
s3.setBucketLifecycleConfiguration(s3BucketName, lifecycleConfig);
```

```
System.out.println("Bucket created and configured.");

// Set the SQS extended client configuration with large payload support enabled.
ExtendedClientConfiguration extendedClientConfig = new ExtendedClientConfiguration()
    .withLargePayloadSupportEnabled(s3, s3BucketName);

AmazonSQS sqsExtended = new AmazonSQSExtendedClient(new AmazonSQSClient(credentials),
 extendedClientConfig);
Region sqsRegion = Region.getRegion(Regions.US_WEST_2);
sqsExtended.setRegion(sqsRegion);
```

L'exemple de code Java suivant crée la fabrique de connexions :

```
// Create the connection factory using the environment variable credential provider.
// Pass the configured Amazon SQS Extended Client to the JMS connection factory.
SQSConnectionFactory connectionFactory = new SQSConnectionFactory(
        new ProviderConfiguration(),
        sqsExtended
        );

// Create the connection.
SQSConnection connection = connectionFactory.createConnection();
```

Exemples Java fonctionnels pour l'utilisation de JMS avec les files d'attente standard Amazon SQS

Les exemples de code suivants montrent comment utiliser le service de messagerie Java (JMS) avec les files d'attente Amazon SQS standard. Pour plus d'informations sur l'utilisation des files d'attente FIFO, consultez Pour créer une file d'attente FIFO, Envoi de messages de façon synchrone et Réception des messages de façon synchrone. (La réception synchrone des messages est identique pour les files d'attente standard et FIFO. Toutefois, les messages des files d'attente FIFO contiennent davantage d'attributs.)

ExampleConfiguration.java

L'exemple de code Java SDK v 1.x suivant définit le nom de file d'attente par défaut, la région et les informations d'identification à utiliser avec les autres exemples Java.

```
/*
 * Copyright 2010-2024 Amazon.com, Inc. or its affiliates. All Rights Reserved.
```

```
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

public class ExampleConfiguration {
    public static final String DEFAULT_QUEUE_NAME = "SQSJMSClientExampleQueue";

    public static final Region DEFAULT_REGION = Region.getRegion(Regions.US_EAST_2);

    private static String getParameter( String args[], int i ) {
        if( i + 1 >= args.length ) {
            throw new IllegalArgumentException( "Missing parameter for " + args[i] );
        }
        return args[i+1];
    }

    /**
     * Parse the command line and return the resulting config. If the config parsing
 fails
     * print the error and the usage message and then call System.exit
     *
     * @param app the app to use when printing the usage string
     * @param args the command line arguments
     * @return the parsed config
     */
    public static ExampleConfiguration parseConfig(String app, String args[]) {
        try {
            return new ExampleConfiguration(args);
        } catch (IllegalArgumentException e) {
            System.err.println( "ERROR: " + e.getMessage() );
            System.err.println();
            System.err.println( "Usage: " + app + " [--queue <queue>] [--region
 <region>] [--credentials <credentials>] ");
            System.err.println( "  or" );
```

```
            System.err.println( "       " + app + " <spring.xml>" );
            System.exit(-1);
            return null;
        }
    }

    private ExampleConfiguration(String args[]) {
        for( int i = 0; i < args.length; ++i ) {
            String arg = args[i];
            if( arg.equals( "--queue" ) ) {
                setQueueName(getParameter(args, i));
                i++;
            } else if( arg.equals( "--region" ) ) {
                String regionName = getParameter(args, i);
                try {
                    setRegion(Region.getRegion(Regions.fromName(regionName)));
                } catch( IllegalArgumentException e ) {
                    throw new IllegalArgumentException( "Unrecognized region " +
regionName );
                }
                i++;
            } else if( arg.equals( "--credentials" ) ) {
                String credsFile = getParameter(args, i);
                try {
                    setCredentialsProvider( new
PropertiesFileCredentialsProvider(credsFile) );
                } catch (AmazonClientException e) {
                    throw new IllegalArgumentException("Error reading credentials from
" + credsFile, e );
                }
                i++;
            } else {
                throw new IllegalArgumentException("Unrecognized option " + arg);
            }
        }
    }

    private String queueName = DEFAULT_QUEUE_NAME;
    private Region region = DEFAULT_REGION;
    private AWSCredentialsProvider credentialsProvider = new
DefaultAWSCredentialsProviderChain();

    public String getQueueName() {
        return queueName;
```

```
    }

    public void setQueueName(String queueName) {
        this.queueName = queueName;
    }

    public Region getRegion() {
        return region;
    }

    public void setRegion(Region region) {
        this.region = region;
    }

    public AWSCredentialsProvider getCredentialsProvider() {
        return credentialsProvider;
    }

    public void setCredentialsProvider(AWSCredentialsProvider credentialsProvider) {
        // Make sure they're usable first
        credentialsProvider.getCredentials();
        this.credentialsProvider = credentialsProvider;
    }
}
```

TextMessageSender.java

L'exemple de code Java suivant crée un producteur de messages texte.

```
/*
 * Copyright 2010-2024 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
```

```
 */

public class TextMessageSender {
    public static void main(String args[]) throws JMSException {
        ExampleConfiguration config =
 ExampleConfiguration.parseConfig("TextMessageSender", args);

        ExampleCommon.setupLogging();

        // Create the connection factory based on the config
        SQSConnectionFactory connectionFactory = new SQSConnectionFactory(
                new ProviderConfiguration(),
                AmazonSQSClientBuilder.standard()
                        .withRegion(config.getRegion().getName())
                        .withCredentials(config.getCredentialsProvider())
                );

        // Create the connection
        SQSConnection connection = connectionFactory.createConnection();

        // Create the queue if needed
        ExampleCommon.ensureQueueExists(connection, config.getQueueName());

        // Create the session
        Session session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);
        MessageProducer producer =
 session.createProducer( session.createQueue( config.getQueueName() ) );

        sendMessages(session, producer);

        // Close the connection. This closes the session automatically
        connection.close();
        System.out.println( "Connection closed" );
    }

    private static void sendMessages( Session session, MessageProducer producer ) {
        BufferedReader inputReader = new BufferedReader(
            new InputStreamReader( System.in, Charset.defaultCharset() ) );

        try {
            String input;
            while( true ) {
                System.out.print( "Enter message to send (leave empty to exit): " );
                input = inputReader.readLine();
```

```
                if( input == null || input.equals("" ) ) break;

                TextMessage message = session.createTextMessage(input);
                producer.send(message);
                System.out.println( "Send message " + message.getJMSMessageID() );
            }
        } catch (EOFException e) {
            // Just return on EOF
        } catch (IOException e) {
            System.err.println( "Failed reading input: " + e.getMessage() );
        } catch (JMSException e) {
            System.err.println( "Failed sending message: " + e.getMessage() );
            e.printStackTrace();
        }
    }
}
```

SyncMessageReceiver.java

L'exemple de code Java suivant crée un consommateur de messages synchrone.

```
/*
 * Copyright 2010-2024 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

public class SyncMessageReceiver {
public static void main(String args[]) throws JMSException {
    ExampleConfiguration config =
 ExampleConfiguration.parseConfig("SyncMessageReceiver", args);

    ExampleCommon.setupLogging();
```

```
    // Create the connection factory based on the config
    SQSConnectionFactory connectionFactory = new SQSConnectionFactory(
            new ProviderConfiguration(),
            AmazonSQSClientBuilder.standard()
                    .withRegion(config.getRegion().getName())
                    .withCredentials(config.getCredentialsProvider())
            );

    // Create the connection
    SQSConnection connection = connectionFactory.createConnection();

    // Create the queue if needed
    ExampleCommon.ensureQueueExists(connection, config.getQueueName());

    // Create the session
    Session session = connection.createSession(false, Session.CLIENT_ACKNOWLEDGE);
    MessageConsumer consumer =
 session.createConsumer( session.createQueue( config.getQueueName() ) );

    connection.start();

    receiveMessages(session, consumer);

    // Close the connection. This closes the session automatically
    connection.close();
    System.out.println( "Connection closed" );
}

private static void receiveMessages( Session session, MessageConsumer consumer ) {
    try {
        while( true ) {
            System.out.println( "Waiting for messages");
            // Wait 1 minute for a message
            Message message = consumer.receive(TimeUnit.MINUTES.toMillis(1));
            if( message == null ) {
                System.out.println( "Shutting down after 1 minute of silence" );
                break;
            }
            ExampleCommon.handleMessage(message);
            message.acknowledge();
            System.out.println( "Acknowledged message " + message.getJMSMessageID() );
        }
    } catch (JMSException e) {
```

```
        System.err.println( "Error receiving from SQS: " + e.getMessage() );
        e.printStackTrace();
    }
}
}
```

AsyncMessageReceiver.java

L'exemple de code Java suivant crée un consommateur de messages asynchrone.

```
/*
 * Copyright 2010-2024 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

public class AsyncMessageReceiver {
    public static void main(String args[]) throws JMSException, InterruptedException {
        ExampleConfiguration config =
 ExampleConfiguration.parseConfig("AsyncMessageReceiver", args);

        ExampleCommon.setupLogging();

        // Create the connection factory based on the config
        SQSConnectionFactory connectionFactory = new SQSConnectionFactory(
                new ProviderConfiguration(),
                AmazonSQSClientBuilder.standard()
                        .withRegion(config.getRegion().getName())
                        .withCredentials(config.getCredentialsProvider())
                );

        // Create the connection
        SQSConnection connection = connectionFactory.createConnection();
```

```
        // Create the queue if needed
        ExampleCommon.ensureQueueExists(connection, config.getQueueName());

        // Create the session
        Session session = connection.createSession(false, Session.CLIENT_ACKNOWLEDGE);
        MessageConsumer consumer =
session.createConsumer( session.createQueue( config.getQueueName() ) );

        // No messages are processed until this is called
        connection.start();

        ReceiverCallback callback = new ReceiverCallback();
        consumer.setMessageListener( callback );

        callback.waitForOneMinuteOfSilence();
        System.out.println( "Returning after one minute of silence" );

        // Close the connection. This closes the session automatically
        connection.close();
        System.out.println( "Connection closed" );
    }


    private static class ReceiverCallback implements MessageListener {
        // Used to listen for message silence
        private volatile long timeOfLastMessage = System.nanoTime();

        public void waitForOneMinuteOfSilence() throws InterruptedException {
            for(;;) {
                long timeSinceLastMessage = System.nanoTime() - timeOfLastMessage;
                long remainingTillOneMinuteOfSilence =
                    TimeUnit.MINUTES.toNanos(1) - timeSinceLastMessage;
                if( remainingTillOneMinuteOfSilence < 0 ) {
                    break;
                }
                TimeUnit.NANOSECONDS.sleep(remainingTillOneMinuteOfSilence);
            }
        }


        @Override
        public void onMessage(Message message) {
            try {
```

```
                ExampleCommon.handleMessage(message);
                message.acknowledge();
                System.out.println( "Acknowledged message " +
 message.getJMSMessageID() );
                timeOfLastMessage = System.nanoTime();
            } catch (JMSException e) {
                System.err.println( "Error processing message: " + e.getMessage() );
                e.printStackTrace();
            }
        }
    }
}
```

SyncMessageReceiverClientAcknowledge.java

L'exemple de code Java suivant crée un consommateur synchrone avec le mode de reconnaissance du client.

```
/*
 * Copyright 2010-2024 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

/**
 * An example class to demonstrate the behavior of CLIENT_ACKNOWLEDGE mode for received
 messages. This example
 * complements the example given in {@link SyncMessageReceiverUnorderedAcknowledge} for
 UNORDERED_ACKNOWLEDGE mode.
 *
 * First, a session, a message producer, and a message consumer are created. Then, two
 messages are sent. Next, two messages
```

```
 * are received but only the second one is acknowledged. After waiting for the
 visibility time out period, an attempt to
 * receive another message is made. It's shown that no message is returned for this
 attempt since in CLIENT_ACKNOWLEDGE mode,
 * as expected, all the messages prior to the acknowledged messages are also
 acknowledged.
 *
 * This ISN'T the behavior for UNORDERED_ACKNOWLEDGE mode. Please see {@link
 SyncMessageReceiverUnorderedAcknowledge}
 * for an example.
 */
public class SyncMessageReceiverClientAcknowledge {

    // Visibility time-out for the queue. It must match to the one set for the queue
 for this example to work.
    private static final long TIME_OUT_SECONDS = 1;

    public static void main(String args[]) throws JMSException, InterruptedException {
        // Create the configuration for the example
        ExampleConfiguration config =
 ExampleConfiguration.parseConfig("SyncMessageReceiverClientAcknowledge", args);

        // Setup logging for the example
        ExampleCommon.setupLogging();

        // Create the connection factory based on the config
        SQSConnectionFactory connectionFactory = new SQSConnectionFactory(
                new ProviderConfiguration(),
                AmazonSQSClientBuilder.standard()
                        .withRegion(config.getRegion().getName())
                        .withCredentials(config.getCredentialsProvider())
                );

        // Create the connection
        SQSConnection connection = connectionFactory.createConnection();

        // Create the queue if needed
        ExampleCommon.ensureQueueExists(connection, config.getQueueName());

        // Create the session  with client acknowledge mode
        Session session = connection.createSession(false, Session.CLIENT_ACKNOWLEDGE);

        // Create the producer and consume
```

```
        MessageProducer producer =
session.createProducer(session.createQueue(config.getQueueName()));
        MessageConsumer consumer =
session.createConsumer(session.createQueue(config.getQueueName()));

        // Open the connection
        connection.start();

        // Send two text messages
        sendMessage(producer, session, "Message 1");
        sendMessage(producer, session, "Message 2");

        // Receive a message and don't acknowledge it
        receiveMessage(consumer, false);

        // Receive another message and acknowledge it
        receiveMessage(consumer, true);

        // Wait for the visibility time out, so that unacknowledged messages reappear
in the queue
        System.out.println("Waiting for visibility timeout...");
        Thread.sleep(TimeUnit.SECONDS.toMillis(TIME_OUT_SECONDS));

        // Attempt to receive another message and acknowledge it. This results in
receiving no messages since
        // we have acknowledged the second message. Although we didn't explicitly
acknowledge the first message,
        // in the CLIENT_ACKNOWLEDGE mode, all the messages received prior to the
explicitly acknowledged message
        // are also acknowledged. Therefore, we have implicitly acknowledged the first
message.
        receiveMessage(consumer, true);

        // Close the connection. This closes the session automatically
        connection.close();
        System.out.println("Connection closed.");
    }

    /**
     * Sends a message through the producer.
     *
     * @param producer Message producer
     * @param session Session
     * @param messageText Text for the message to be sent
```

```
     * @throws JMSException
     */
    private static void sendMessage(MessageProducer producer, Session session, String
 messageText) throws JMSException {
        // Create a text message and send it
        producer.send(session.createTextMessage(messageText));
    }

    /**
     * Receives a message through the consumer synchronously with the default timeout
 (TIME_OUT_SECONDS).
     * If a message is received, the message is printed. If no message is received,
 "Queue is empty!" is
     * printed.
     *
     * @param consumer Message consumer
     * @param acknowledge If true and a message is received, the received message is
 acknowledged.
     * @throws JMSException
     */
    private static void receiveMessage(MessageConsumer consumer, boolean acknowledge)
 throws JMSException {
        // Receive a message
        Message message =
 consumer.receive(TimeUnit.SECONDS.toMillis(TIME_OUT_SECONDS));

        if (message == null) {
            System.out.println("Queue is empty!");
        } else {
            // Since this queue has only text messages, cast the message object and
 print the text
            System.out.println("Received: " + ((TextMessage) message).getText());

            // Acknowledge the message if asked
            if (acknowledge) message.acknowledge();
        }
    }
}
```

SyncMessageReceiverUnorderedAcknowledge.java

L'exemple de code Java suivant crée un consommateur synchrone avec le mode de reconnaissance sans ordre de réception.

```
/*
 * Copyright 2010-2024 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

/**
 * An example class to demonstrate the behavior of UNORDERED_ACKNOWLEDGE mode for
 received messages. This example
 * complements the example given in {@link SyncMessageReceiverClientAcknowledge} for
 CLIENT_ACKNOWLEDGE mode.
 *
 * First, a session, a message producer, and a message consumer are created. Then, two
 messages are sent. Next, two messages
 * are received but only the second one is acknowledged. After waiting for the
 visibility time out period, an attempt to
 * receive another message is made. It's shown that the first message received in the
 prior attempt is returned again
 * for the second attempt. In UNORDERED_ACKNOWLEDGE mode, all the messages must be
 explicitly acknowledged no matter what
 * the order they're received.
 *
 * This ISN'T the behavior for CLIENT_ACKNOWLEDGE mode. Please see {@link
 SyncMessageReceiverClientAcknowledge}
 * for an example.
 */
public class SyncMessageReceiverUnorderedAcknowledge {

    // Visibility time-out for the queue. It must match to the one set for the queue
 for this example to work.
    private static final long TIME_OUT_SECONDS = 1;

    public static void main(String args[]) throws JMSException, InterruptedException {
```

```
        // Create the configuration for the example
        ExampleConfiguration config =
ExampleConfiguration.parseConfig("SyncMessageReceiverUnorderedAcknowledge", args);

        // Setup logging for the example
        ExampleCommon.setupLogging();

        // Create the connection factory based on the config
        SQSConnectionFactory connectionFactory = new SQSConnectionFactory(
                new ProviderConfiguration(),
                AmazonSQSClientBuilder.standard()
                        .withRegion(config.getRegion().getName())
                        .withCredentials(config.getCredentialsProvider())
                );

        // Create the connection
        SQSConnection connection = connectionFactory.createConnection();

        // Create the queue if needed
        ExampleCommon.ensureQueueExists(connection, config.getQueueName());

        // Create the session  with unordered acknowledge mode
        Session session = connection.createSession(false,
SQSSession.UNORDERED_ACKNOWLEDGE);

        // Create the producer and consume
        MessageProducer producer =
session.createProducer(session.createQueue(config.getQueueName()));
        MessageConsumer consumer =
session.createConsumer(session.createQueue(config.getQueueName()));

        // Open the connection
        connection.start();

        // Send two text messages
        sendMessage(producer, session, "Message 1");
        sendMessage(producer, session, "Message 2");

        // Receive a message and don't acknowledge it
        receiveMessage(consumer, false);

        // Receive another message and acknowledge it
        receiveMessage(consumer, true);
```

```
        // Wait for the visibility time out, so that unacknowledged messages reappear
in the queue
        System.out.println("Waiting for visibility timeout...");
        Thread.sleep(TimeUnit.SECONDS.toMillis(TIME_OUT_SECONDS));

        // Attempt to receive another message and acknowledge it. This results in
receiving the first message since
        // we have acknowledged only the second message. In the UNORDERED_ACKNOWLEDGE
mode, all the messages must
        // be explicitly acknowledged.
        receiveMessage(consumer, true);

        // Close the connection. This closes the session automatically
        connection.close();
        System.out.println("Connection closed.");
    }

    /**
     * Sends a message through the producer.
     *
     * @param producer Message producer
     * @param session Session
     * @param messageText Text for the message to be sent
     * @throws JMSException
     */
    private static void sendMessage(MessageProducer producer, Session session, String
messageText) throws JMSException {
        // Create a text message and send it
        producer.send(session.createTextMessage(messageText));
    }

    /**
     * Receives a message through the consumer synchronously with the default timeout
(TIME_OUT_SECONDS).
     * If a message is received, the message is printed. If no message is received,
"Queue is empty!" is
     * printed.
     *
     * @param consumer Message consumer
     * @param acknowledge If true and a message is received, the received message is
acknowledged.
     * @throws JMSException
     */
```

```
    private static void receiveMessage(MessageConsumer consumer, boolean acknowledge)
 throws JMSException {
        // Receive a message
        Message message =
 consumer.receive(TimeUnit.SECONDS.toMillis(TIME_OUT_SECONDS));

        if (message == null) {
            System.out.println("Queue is empty!");
        } else {
            // Since this queue has only text messages, cast the message object and
 print the text
            System.out.println("Received: " + ((TextMessage) message).getText());

            // Acknowledge the message if asked
            if (acknowledge) message.acknowledge();
        }
    }
}
```

SpringExampleConfiguration.xml

L'exemple de code XML suivant est un fichier de configuration bean pour SpringExample.java.

```
<!--
    Copyright 2010-2024 Amazon.com, Inc. or its affiliates. All Rights Reserved.

    Licensed under the Apache License, Version 2.0 (the "License").
    You may not use this file except in compliance with the License.
    A copy of the License is located at

    https://aws.amazon.com/apache2.0

    or in the "license" file accompanying this file. This file is distributed
    on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
    express or implied. See the License for the specific language governing
    permissions and limitations under the License.
-->

<?xml version="1.0" encoding="UTF-8"?>
<beans
    xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:util="http://www.springframework.org/schema/util"
```

```
    xmlns:p="http://www.springframework.org/schema/p"
    xsi:schemaLocation="
        http://www.springframework.org/schema/beans http://www.springframework.org/
schema/beans/spring-beans-3.0.xsd
        http://www.springframework.org/schema/util http://www.springframework.org/
schema/util/spring-util-3.0.xsd
    ">

    <bean id="CredentialsProviderBean"
 class="com.amazonaws.auth.DefaultAWSCredentialsProviderChain"/>

    <bean id="ClientBuilder" class="com.amazonaws.services.sqs.AmazonSQSClientBuilder"
 factory-method="standard">
        <property name="region" value="us-east-2"/>
        <property name="credentials" ref="CredentialsProviderBean"/>
    </bean>

    <bean id="ProviderConfiguration"
 class="com.amazon.sqs.javamessaging.ProviderConfiguration">
        <property name="numberOfMessagesToPrefetch" value="5"/>
    </bean>

    <bean id="ConnectionFactory"
 class="com.amazon.sqs.javamessaging.SQSConnectionFactory">
        <constructor-arg ref="ProviderConfiguration" />
        <constructor-arg ref="ClientBuilder" />
    </bean>

    <bean id="Connection" class="javax.jms.Connection"
        factory-bean="ConnectionFactory"
        factory-method="createConnection"
        init-method="start"
        destroy-method="close" />

    <bean id="QueueName" class="java.lang.String">
        <constructor-arg value="SQSJMSClientExampleQueue"/>
    </bean>
</beans>
```

SpringExample.java

L'exemple de code Java suivant utilise le fichier de configuration bean pour initialiser vos objets.

```
/*
 * Copyright 2010-2024 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

public class SpringExample {
    public static void main(String args[]) throws JMSException {
        if( args.length != 1 || !args[0].endsWith(".xml")) {
            System.err.println( "Usage: " + SpringExample.class.getName() + " <spring
 config.xml>" );
            System.exit(1);
        }

        File springFile = new File( args[0] );
        if( !springFile.exists() || !springFile.canRead() ) {
            System.err.println( "File " + args[0] + " doesn't exist or isn't
 readable.");
            System.exit(2);
        }

        ExampleCommon.setupLogging();

        FileSystemXmlApplicationContext context =
            new FileSystemXmlApplicationContext( "file://" +
 springFile.getAbsolutePath() );

        Connection connection;
        try {
            connection = context.getBean(Connection.class);
        } catch( NoSuchBeanDefinitionException e ) {
            System.err.println( "Can't find the JMS connection to use: " +
 e.getMessage() );
```

```
            System.exit(3);
            return;
        }

        String queueName;
        try {
            queueName = context.getBean("QueueName", String.class);
        } catch( NoSuchBeanDefinitionException e ) {
            System.err.println( "Can't find the name of the queue to use: " +
e.getMessage() );
            System.exit(3);
            return;
        }

        if( connection instanceof SQSConnection ) {
            ExampleCommon.ensureQueueExists( (SQSConnection) connection, queueName );
        }

        // Create the session
        Session session = connection.createSession(false, Session.CLIENT_ACKNOWLEDGE);
        MessageConsumer consumer =
session.createConsumer( session.createQueue( queueName) );

        receiveMessages(session, consumer);

        // The context can be setup to close the connection for us
        context.close();
        System.out.println( "Context closed" );
    }

    private static void receiveMessages( Session session, MessageConsumer consumer ) {
        try {
            while( true ) {
                System.out.println( "Waiting for messages");
                // Wait 1 minute for a message
                Message message = consumer.receive(TimeUnit.MINUTES.toMillis(1));
                if( message == null ) {
                    System.out.println( "Shutting down after 1 minute of silence" );
                    break;
                }
                ExampleCommon.handleMessage(message);
                message.acknowledge();
                System.out.println( "Acknowledged message" );
            }
```

```
        } catch (JMSException e) {
            System.err.println( "Error receiving from SQS: " + e.getMessage() );
            e.printStackTrace();
        }
    }
}
```

ExampleCommon.java

L'exemple de code Java suivant vérifie s'il existe une file d'attente Amazon SQS, puis en crée une si ce n'est pas le cas. Il comprend également un exemple de code de journalisation.

```
/*
 * Copyright 2010-2024 Amazon.com, Inc. or its affiliates. All Rights Reserved.
 *
 * Licensed under the Apache License, Version 2.0 (the "License").
 * You may not use this file except in compliance with the License.
 * A copy of the License is located at
 *
 *  https://aws.amazon.com/apache2.0
 *
 * or in the "license" file accompanying this file. This file is distributed
 * on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
 * express or implied. See the License for the specific language governing
 * permissions and limitations under the License.
 *
 */

public class ExampleCommon {
    /**
     * A utility function to check the queue exists and create it if needed. For most
     * use cases this is usually done by an administrator before the application is
 run.
     */
    public static void ensureQueueExists(SQSConnection connection, String queueName)
 throws JMSException {
        AmazonSQSMessagingClientWrapper client =
 connection.getWrappedAmazonSQSClient();

        /**
         * In most cases, you can do this with just a createQueue call, but
 GetQueueUrl
```

```
         * (called by queueExists) is a faster operation for the common case where the
 queue
         * already exists. Also many users and roles have permission to call
 GetQueueUrl
         * but don't have permission to call CreateQueue.
         */
        if( !client.queueExists(queueName) ) {
            client.createQueue( queueName );
        }
    }

    public static void setupLogging() {
        // Setup logging
        BasicConfigurator.configure();
        Logger.getRootLogger().setLevel(Level.WARN);
    }

    public static void handleMessage(Message message) throws JMSException {
        System.out.println( "Got message " + message.getJMSMessageID() );
        System.out.println( "Content: ");
        if( message instanceof TextMessage ) {
            TextMessage txtMessage = ( TextMessage ) message;
            System.out.println( "\t" + txtMessage.getText() );
        } else if( message instanceof BytesMessage ){
            BytesMessage byteMessage = ( BytesMessage ) message;
            // Assume the length fits in an int - SQS only supports sizes up to 256k so
 that
            // should be true
            byte[] bytes = new byte[(int)byteMessage.getBodyLength()];
            byteMessage.readBytes(bytes);
            System.out.println( "\t" +  Base64.encodeAsString( bytes ) );
        } else if( message instanceof ObjectMessage ) {
            ObjectMessage objMessage = (ObjectMessage) message;
            System.out.println( "\t" + objMessage.getObject() );
        }
    }
}
```

Implémentations de JMS 1.1 prises en charge par Amazon SQS

La bibliothèque de messagerie Java Amazon SQS prend en charge les implémentations JMS 1.1 suivantes. Pour plus d'informations sur les fonctionnalités prises en charge de la bibliothèque de messagerie Java Amazon SQS, consultez la Foire aux questions sur Amazon SQS.

Interfaces courantes prises en charge

- `Connection`
- `ConnectionFactory`
- `Destination`
- `Session`
- `MessageConsumer`
- `MessageProducer`

Types de messages pris en charge

- `ByteMessage`
- `ObjectMessage`
- `TextMessage`

Modes de reconnaissance des messages pris en charge

- `AUTO_ACKNOWLEDGE`
- `CLIENT_ACKNOWLEDGE`
- `DUPS_OK_ACKNOWLEDGE`
- `UNORDERED_ACKNOWLEDGE`

> **Note**
>
> Le mode `UNORDERED_ACKNOWLEDGE` ne fait pas partie de la spécification JMS 1.1. Ce mode permet à Amazon SQS d'autoriser un client JMS à accepter explicitement un message.

En-têtes définis par JMS et propriétés réservées

Pour l'envoi de messages

Lorsque vous envoyez des messages, vous pouvez définir les en-têtes et propriétés ci-après pour chaque message :

- `JMSXGroupID` (obligatoire pour les files d'attente FIFO, non autorisé pour les files d'attente standard)
- `JMS_SQS_DeduplicationId` (facultatif pour les files d'attente FIFO, non autorisé pour les files d'attente standard)

Après l'envoi de messages, Amazon SQS définit les en-têtes et propriétés ci-après pour chaque message :

- `JMSMessageID`
- `JMS_SQS_SequenceNumber` (uniquement pour les files d'attente FIFO)

Pour la réception de messages

Lorsque vous recevez des messages, Amazon SQS définit les en-têtes et propriétés ci-après pour chaque message :

- `JMSDestination`
- `JMSMessageID`
- `JMSRedelivered`
- `JMSXDeliveryCount`
- `JMSXGroupID` (uniquement pour les files d'attente FIFO)
- `JMS_SQS_DeduplicationId` (uniquement pour les files d'attente FIFO)
- `JMS_SQS_SequenceNumber` (uniquement pour les files d'attente FIFO)

Didacticiels Amazon SQS

Cette section fournit des didacticiels que vous pouvez utiliser pour explorer les fonctions et fonctionnalités d'Amazon SQS.

Rubriques

Création d'une file d'attente Amazon SQS à l'aide de AWS CloudFormation

Vous pouvez utiliser la AWS CloudFormation console et un modèle JSON (ou YAML) pour créer une file d'attente Amazon SQS. Pour plus d'informations, consultez Utiliser des modèles AWS CloudFormation et la Ressource `AWS::SQS::Queue` dans le Guide de l'utilisateur AWS CloudFormation .

À utiliser AWS CloudFormation pour créer une file d'attente Amazon SQS.

1. Copiez le code JSON suivant dans un fichier nommé `MyQueue.json`. Pour créer une file d'attente standard, omettez les propriétés `FifoQueue` et `ContentBasedDeduplication`. Pour plus d'informations sur la déduplication basée sur le contenu, consultez Traitement effectué en une seule fois dans Amazon SQS.

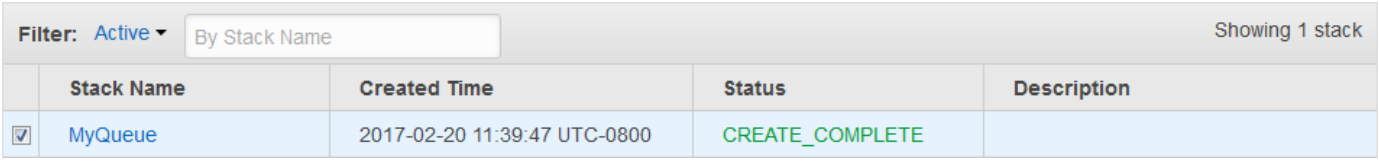
   > **Note**
   >
   > Le nom d'une file d'attente FIFO doit se terminer par le suffixe `.fifo`.

   ```
   {
      "AWSTemplateFormatVersion": "2010-09-09",
      "Resources": {
         "MyQueue": {
            "Properties": {
               "QueueName": "MyQueue.fifo",
   ```

```
            "FifoQueue": true,
            "ContentBasedDeduplication": true
             },
         "Type": "AWS::SQS::Queue"
         }
      },
   "Outputs": {
      "QueueName": {
         "Description": "The name of the queue",
         "Value": {
            "Fn::GetAtt": [
               "MyQueue",
               "QueueName"
            ]
         }
      },
      "QueueURL": {
         "Description": "The URL of the queue",
         "Value": {
            "Ref": "MyQueue"
         }
      },
      "QueueARN": {
         "Description": "The ARN of the queue",
         "Value": {
            "Fn::GetAtt": [
               "MyQueue",
               "Arn"
            ]
         }
      }
   }
}
```

2. Connectez-vous à la console AWS CloudFormation, puis choisissez Créer une pile.
3. Dans le panneau Spécifier un modèle, choisissez Charger un fichier modèle, choisissez votre `MyQueue.json` fichier, puis cliquez sur Suivant.
4. Sur la page Spécifier les détails, tapez `MyQueue` pour Nom de la pile, puis choisissez Suivant.
5. Dans la page Options, choisissez Next (Suivant).
6. Sur la page Review (Vérification), choisissez Create (Créer).

AWS CloudFormation commence à créer la `MyQueue` pile et affiche le statut CREATE_IN_PROGRESS. Lorsque le processus est terminé, AWS CloudFormation affiche l'état CREATE_COMPLETE.



7. (Facultatif) Pour afficher le nom, l'URL et l'ARN de la file d'attente, choisissez le nom de la pile, puis, sur la page suivante, développez la section Outputs.

Didacticiel : envoi d'un message à une file d'attente Amazon SQS à partir d'Amazon Virtual Private Cloud

Dans ce didacticiel, vous apprenez à envoyer des messages à une file d'attente Amazon SQS sur un réseau privé et sécurisé. Ce réseau est composé d'un VPC qui contient une instance Amazon EC2. L'instance se connecte à Amazon SQS via le point de terminaison d'un VPC d'interface, ce qui vous permet de vous connecter à l'instance Amazon EC2 et d'envoyer des messages à la file d'attente Amazon SQS, même si le réseau est déconnecté de l'Internet public. Pour plus d'informations, consultez Points de terminaison Amazon Virtual Private Cloud pour Amazon SQS.

Important

- Vous pouvez utiliser Amazon Virtual Private Cloud uniquement avec des points de terminaison HTTPS Amazon SQS.
- Lorsque vous configurez Amazon SQS pour que les messages soient envoyés depuis Amazon VPC, vous devez activer le DNS privé et spécifier les points de terminaison au format `sqs.us-east-2.amazonaws.com`.
- Le DNS privé ne prend pas en charge les points de terminaison existants, tels que `queue.amazonaws.com` ou `us-east-2.queue.amazonaws.com`.

Rubriques

Étape 1 : Créer une paire de clés Amazon EC2

Une paire de clés vous permet de vous connecter à une instance Amazon EC2. Elle se compose d'une clé publique qui chiffre vos informations de connexion, et d'une clé privée qui déchiffre ces informations.

1. Connectez-vous à la console Amazon EC2.
2. Dans le menu de navigation, sous Réseau et sécurité, choisissez Paires de clés.
3. Choisissez Create Key Pair (Créer une paire de clés).
4. Dans la boîte de dialogue Créer une paire de clés, pour Nom de la paire de clés, entrez `SQS-VPCE-Tutorial-Key-Pair`, puis choisissez Créer.
5. Votre navigateur télécharge automatiquement le fichier de clé privée `SQS-VPCE-Tutorial-Key-Pair.pem`.

   > **Important**
   >
   > Enregistrez ce fichier dans un emplacement sûr. EC2 ne génère pas de fichier `.pem` pour la même paire de clés une deuxième fois.

6. Pour autoriser un client SSH à se connecter à votre instance EC2, définissez les autorisations de votre fichier de clé privée de sorte que seul votre utilisateur soit autorisé à lire le fichier. Par exemple :

   ```
   chmod 400 SQS-VPCE-Tutorial-Key-Pair.pem
   ```

Étape 2 : Création de AWS ressources

Pour configurer l'infrastructure nécessaire, vous devez utiliser un AWS CloudFormation modèle, qui est un modèle pour créer une pile composée de AWS ressources, telles que des instances Amazon EC2 et des files d'attente Amazon SQS.

La pile pour ce didacticiel inclut les ressources suivantes :

- Un VPC et les ressources de mise en réseau associées, notamment un sous-réseau, un groupe de sécurité, une passerelle Internet et une table de routage
- Une instance Amazon EC2 lancée dans le sous-réseau VPC
- Une file d'attente Amazon SQS

1. Téléchargez le AWS CloudFormation modèle nommé `SQS-VPCE-Tutorial-CloudFormation.yaml`deGitHub.
2. Connectez-vous à la console AWS CloudFormation.
3. Sélectionnez Créer une pile.
4. Sur la page Sélectionner un modèle, choisissez Télécharger un modèle sur Amazon S3, sélectionnez le fichier `SQS-VPCE-SQS-Tutorial-CloudFormation.yaml`, puis choisissez Suivant.
5. Sur la page Spécification de détails de base de données, procédez comme suit :
   a. Dans le champ Nom de la pile, saisissez `SQS-VPCE-Tutorial-Stack`.
   b. Pour KeyName, choisissez SQS-VPCE-Tutorial-Key-Pair.
   c. Choisissez Suivant.
6. Dans la page Options, choisissez Suivant.
7. Sur la page de révision, dans la section Fonctionnalités, choisissez Je reconnais que cela AWS CloudFormation pourrait créer des ressources IAM avec des noms personnalisés. , puis choisissez Create.

AWS CloudFormation commence à créer la pile et affiche le statut CREATE_IN_PROGRESS. Lorsque le processus est terminé, AWS CloudFormation affiche l'état CREATE_COMPLETE.

Étape 3 : Confirmer que votre instance EC2 n'est pas accessible publiquement

Votre AWS CloudFormation modèle lance une instance EC2 nommée `SQS-VPCE-Tutorial-EC2-Instance` dans votre VPC. Cette instance EC2 n'autorise pas le trafic sortant et n'est pas capable d'envoyer des messages à Amazon SQS. Pour le vérifier, vous devez vous connecter à l'instance, essayer de vous connecter à un point de terminaison public, puis essayer d'envoyer un message à Amazon SQS.

1. Connectez-vous à la console Amazon EC2.

2. Dans le menu de navigation, sous Instances, choisissez Instances.
3. Sélectionnez SQS-VPCE-. Tutorial-EC2Instance
4. Copiez le nom d'hôte sous DNS public (IPv4). Par exemple, ec2-203-0-113-0.us-west-2.compute.amazonaws.com.
5. À partir du répertoire contenant la paire de clés que vous venez de créer, connectez-vous à l'instance à l'aide de la commande suivante. Par exemple :

```
ssh -i SQS-VPCE-Tutorial-Key-Pair.pem ec2-user@ec2-203-0-113-0.us-east-2.compute.amazonaws.com
```

6. Essayez de vous connecter à n'importe quel point de terminaison public. Par exemple :

```
ping amazon.com
```

   La tentative de connexion échoue, comme prévu.

7. Connectez-vous à la console Amazon SQS.
8. Dans la liste des files d'attente, sélectionnez la file créée par votre AWS CloudFormation modèle, par exemple, VPCE-SQS-Tutorial-Stack-CFQueue-1abcdefgh2ijk.
9. Dans le tableau Détails, copiez l'URL. Par exemple, https://sqs.us-east-2.amazonaws.com/123456789012/.
10. À partir de votre instance EC2, essayez de publier un message dans la file d'attente à l'aide de la commande suivante. Par exemple :

```
aws sqs send-message --region us-east-2 --endpoint-url https://sqs.us-east-2.amazonaws.com/ --queue-url https://sqs.us-east-2.amazonaws.com/123456789012/
 --message-body "Hello from Amazon SQS."
```

    La tentative d'envoi échoue, comme prévu.

> **Important**
>
> Plus tard, lorsque vous créerez le point de terminaison d'un VPC pour Amazon SQS, votre tentative d'envoi réussira.

Étape 4 : Création du point de terminaison d'un VPC Amazon pour Amazon SQS

Pour connecter votre VPC à Amazon SQS, vous devez définir le point de terminaison d'un VPC d'interface. Après avoir ajouté le point de terminaison, vous pouvez utiliser l'API Amazon SQS à partir de l'instance EC2 dans votre VPC. Cela vous permet d'envoyer des messages à une file d'attente du AWS réseau sans passer par l'Internet public.

> **Note**
>
> L'instance EC2 n'a toujours pas accès aux autres AWS services et points de terminaison sur Internet.

1. Connectez-vous à la console Amazon VPC.
2. Dans le menu de navigation, choisissez Points de terminaison.
3. Choisissez Créer un point de terminaison.
4. Sur la page Créer un point de terminaison, pour Nom du service, choisissez le nom de service pour Amazon SQS.

   > **Note**
   >
   > Les noms des services varient en fonction de la AWS région actuelle. Par exemple, si vous vous trouvez dans la région USA Est (Ohio), le nom du service est com.amazonaws.***us-east-2***.sqs.

5. Pour VPC, choisissez SQS-VPCE-Tutorial-VPC.
6. Pour Sous-réseaux, choisissez le sous-réseau dont l'ID de sous-réseau contient SQS-VPCE-Tutorial-Subnet.
7. Pour Groupe de sécurité, choisissez Select security groups (Sélectionner des groupes de sécurité), puis choisissez le groupe de sécurité dont le Nom du groupe contient SQS VPCE Tutorial Security Group.
8. Choisissez Créer un point de terminaison.

   Le point de terminaison de VPC d'interface est créé et son ID s'affiche. Par exemple, vpce-0ab1cdef2ghi3j456k.

9. Choisissez Fermer.

   La console Amazon VPC ouvre la page Points de terminaison.

Amazon VPC commence à créer le point de terminaison et affiche le statut en suspens. Lorsque le processus est terminé, Amazon VPC affiche le statut disponible.

Étape 5 : Envoyer un message à votre file d'attente Amazon SQS

Maintenant que votre VPC inclut un point de terminaison pour Amazon SQS, vous pouvez vous connecter à votre instance EC2 et envoyer des messages à votre file d'attente.

1. Reconnectez-vous à votre instance EC2. Par exemple :

   ```
   ssh -i SQS-VPCE-Tutorial-Key-Pair.pem ec2-user@ec2-203-0-113-0.us-
   east-2.compute.amazonaws.com
   ```

2. Essayez de publier à nouveau un message dans la file d'attente à l'aide de la commande suivante. Par exemple :

   ```
   aws sqs send-message --region us-east-2 --endpoint-url https://sqs.us-
   east-2.amazonaws.com/ --queue-url https://sqs.us-east-2.amazonaws.com/123456789012/
    --message-body "Hello from Amazon SQS."
   ```

   La tentative d'envoi aboutit et la valeur de hachage MD5 du corps de message et l'ID de message s'affichent. Par exemple :

   ```
   {
    "MD5OfMessageBody": "a1bcd2ef3g45hi678j90klmn12p34qr5",
    "MessageId": "12345a67-8901-2345-bc67-d890123e45fg"
   }
   ```

Pour plus d'informations sur la réception et la suppression du message de la file d'attente créée par votre AWS CloudFormation modèle (par exemple, vpce-sqs-tutorial-stack-cfqueue-1abcdefgh2ijk), consultez. Réception et suppression d'un message dans Amazon SQS

Pour plus d'informations sur la suppression de vos ressources, consultez ce qui suit :

- Suppression du point de terminaison d'un VPC dans le Guide de l'utilisateur Amazon VPC

- Supprimer une file d'attente Amazon SQS
- Mettez fin à votre instance dans le guide de l'utilisateur Amazon EC2
- Suppression de votre VPC dans le Guide de l'utilisateur Amazon VPC
- Supprimer une pile sur la AWS CloudFormation console dans le guide de AWS CloudFormation l'utilisateur
- Supprimer votre paire de clés dans le guide de l'utilisateur Amazon EC2

Résolution des problèmes dans Amazon SQS

Les rubriques suivantes fournissent des conseils de résolution des problèmes et erreurs courants que vous pouvez rencontrer lors de l'utilisation de la console Amazon SQS, de l'API Amazon SQS ou d'autres outils avec Amazon SQS. Si vous rencontrez un problème qui n'est pas répertorié ici, vous pouvez utiliser le bouton Commentaire sur cette page pour le signaler.

Pour plus de conseils de dépannage et de réponses aux questions courantes de support, visitez le Centre de connaissances AWS.

Rubriques

Résoudre une erreur de refus d'accès dans Amazon SQS

Les rubriques suivantes présentent les causes `AccessDenied` ou les `AccessDeniedException` erreurs les plus courantes des appels d'API Amazon SQS. Pour plus d'informations sur la résolution de ces erreurs, consultez Comment résoudre les erreurs « » ou « AccessDenied AccessDenied Exception » lors des appels d'API Amazon SQS ? dans le guide du centre de AWS connaissances.

Exemples de messages d'erreur :

```
An error occurred (AccessDenied) when calling the SendMessage operation: Access to
        the resource https://sqs.us-east-1.amazonaws.com/ is denied.
```

- ou -

```
An error occurred (KMS.AccessDeniedException) when calling the SendMessage
```

```
        operation: User: arn:aws:iam::xxxxx:user/xxxx is not authorized to perform:
        kms:GenerateDataKey on resource: arn:aws:kms:us-east-1:xxxx:key/xxxx with an
explicit
        deny.
```

Rubriques

- Politique de file d'attente Amazon SQS et politique IAM
- AWS Key Management Service autorisations
- Politique de point de terminaison d'un VPC
- Politique de contrôle des services de l'organisation

Politique de file d'attente Amazon SQS et politique IAM

Pour vérifier si le demandeur dispose des autorisations appropriées pour effectuer une opération Amazon SQS, procédez comme suit :

- Identifiez le principal IAM qui effectue l'appel d'API Amazon SQS. Si le principal IAM possède le même compte, la politique de file d'attente Amazon SQS ou AWS la politique Identity and Access Management (IAM) doivent inclure des autorisations permettant d'autoriser explicitement l'accès à l'action.
- Si le principal est une entité IAM :
  - Vous pouvez identifier votre utilisateur ou votre rôle IAM en cochant le coin supérieur droit du AWS Management Console ou en utilisant la commande. `aws sts get-caller-identity`
  - Vérifiez les politiques IAM attachées à l'utilisateur ou au rôle IAM. Vous pouvez choisir l'une des méthodes suivantes :
    - Testez les politiques IAM avec le simulateur de politiques IAM.
    - Passez en revue les différents types de politiques IAM.
  - Si nécessaire, modifiez votre politique d'utilisateur IAM.
  - Vérifiez la politique de file d'attente et modifiez-la si nécessaire.
- Si le principal est un AWS service, la politique de file d'attente Amazon SQS doit autoriser explicitement l'accès.
- Si le principal est un principal multicompte, la politique de file d'attente Amazon SQS et la politique IAM doivent autoriser explicitement l'accès.
- Si la politique utilise un élément de condition, vérifiez que la condition restreint l'accès.

> **Important**
>
> Un refus explicite dans l'une ou l'autre des politiques remplace une autorisation explicite. Voici quelques exemples de base des politiques Amazon SQS.

AWS Key Management Service autorisations

Si le chiffrement côté serveur (SSE) de votre file d'attente Amazon SQS est activé et qu'un client est géré AWS KMS key, les autorisations doivent être accordées à la fois aux producteurs et aux consommateurs. Pour vérifier si une file d'attente est chiffrée, vous pouvez utiliser l'`KmsMasterKeyId`attribut `GetQueueAttributes`API ou depuis la console de file d'attente sous Chiffrement.

- Autorisations requises pour les producteurs :

```
{
"Effect": "Allow",
"Action": [
    "kms:GenerateDataKey",
    "kms:Decrypt"
],
"Resource": "<Key ARN>"
}
```

- Autorisations requises pour les consommateurs :

```
{
"Effect": "Allow",
"Action": [
    "kms:Decrypt"
],
"Resource": "<Key ARN>"
}
```

- Autorisations requises pour l'accès entre comptes :

```
{
"Effect": "Allow",
"Action": [
    "kms:DescribeKey",
```

```
    "kms:Decrypt",
    "kms:ReEncrypt",
    "kms:GenerateDataKey"
],
"Resource": "<Key ARN>"
}
```

Vous pouvez utiliser l'une des méthodes suivantes pour activer le chiffrement d'une file d'attente Amazon SQS :

- SSE-Amazon SQS (clé de chiffrement créée et gérée par le service Amazon SQS.)
- AWS clé par défaut gérée (alias/aws/sqs)
- Clé gérée par le client

Toutefois, si vous utilisez une clé KMS AWS gérée, vous ne pouvez pas modifier la politique de clé par défaut. Par conséquent, pour donner accès à d'autres services et à des comptes croisés, utilisez une clé gérée par le client. Cela vous permet de modifier la politique clé.

Politique de point de terminaison d'un VPC

Si vous accédez à Amazon SQS via un point de terminaison Amazon Virtual Private Cloud (Amazon VPC), la politique de point de terminaison Amazon SQS VPC doit autoriser l'accès. Vous pouvez créer une politique pour les points de terminaison Amazon VPC pour Amazon SQS, dans laquelle vous pouvez spécifier les éléments suivants :

1. Le principal qui peut exécuter des actions.
2. Les actions qui peuvent être effectuées.
3. Les ressources sur lesquelles les actions peuvent être exécutées.

Dans l'exemple suivant, la politique de point de terminaison VPC indique que l'utilisateur IAM *MyUser*est autorisé à envoyer des messages à la file d'attente Amazon SQS. *MyQueue* L'accès aux autres actions, aux utilisateurs IAM et aux ressources Amazon SQS est refusé via le point de terminaison VPC.

```
{
   "Statement": [{
      "Action": ["sqs:SendMessage"],
```

```
      "Effect": "Allow",
      "Resource": "arn:aws:sqs:us-east-2:123456789012:MyQueue",
      "Principal": {
        "AWS": "arn:aws:iam:123456789012:user/MyUser"
      }
   }]
}
```

Politique de contrôle des services de l'organisation

Si vous Compte AWS appartenez à une organisation, AWS Organizations les politiques peuvent vous empêcher d'accéder à vos files d'attente Amazon SQS. Par défaut, AWS Organizations les politiques ne bloquent aucune demande adressée à Amazon SQS. Assurez-vous toutefois que vos AWS Organizations politiques n'ont pas été configurées pour bloquer l'accès aux files d'attente Amazon SQS. Pour savoir comment vérifier vos AWS Organizations politiques, consultez la section Liste de toutes les politiques dans le guide de AWS Organizations l'utilisateur.

Résoudre les erreurs d'API Amazon SQS

Les rubriques suivantes présentent les erreurs les plus courantes renvoyées lors des appels d'API Amazon SQS et expliquent comment les résoudre.

Rubriques

- QueueDoesNotExist erreur
- InvalidAttributeValue erreur
- ReceiptHandle erreur

QueueDoesNotExist erreur

Cette erreur sera renvoyée lorsque le service Amazon SQS ne trouvera pas la file d'attente mentionnée pour l'action Amazon SQS.

Causes possibles et mesures d'atténuation :

- Région incorrecte : vérifiez la configuration du client Amazon SQS pour vérifier que vous avez configuré la bonne région sur le client. Lorsque vous ne configurez pas de région sur le client, le SDK AWS CLI choisit la région dans le fichier de configuration ou dans la variable

d'environnement. Si le SDK ne trouve aucune région dans le fichier de configuration, il définit la région sur us-east-1 par défaut.

- La file d'attente a peut-être été supprimée récemment : si la file d'attente a été supprimée avant que l'appel d'API ne soit effectué, l'appel d'API renverra cette erreur. Vérifiez toutes CloudTrail les DeleteQueueopérations effectuées avant le moment où l'erreur s'est produite.
- Problèmes d'autorisation : si l'utilisateur ou le rôle demandeur AWS Identity and Access Management (IAM) ne dispose pas des autorisations requises, le message d'erreur suivant peut s'afficher :

```
The specified queue does not exist or you do not have access to it.
```

Vérifiez les autorisations et effectuez l'appel d'API avec les autorisations correctes.

Pour plus de détails sur la résolution de l'`QueueDoesNotExist`erreur, consultez Comment résoudre l' QueueDoesNotExist erreur lorsque je passe des appels d'API à ma file d'attente Amazon SQS ? dans le guide du centre de AWS connaissances.

InvalidAttributeValue erreur

Cette erreur sera renvoyée lors de la mise à jour de la politique de ressources de file d'attente Amazon SQS ou des propriétés avec une politique ou un principal incorrect.

Causes possibles et mesures d'atténuation :

- Politique de ressources non valide : vérifiez que la politique de ressources contient tous les champs obligatoires. Pour plus d'informations, consultez les sections Référence aux éléments de stratégie IAM JSON et Validation des politiques IAM. Vous pouvez également utiliser le générateur de politiques IAM pour créer et tester une politique de ressources Amazon SQS. Assurez-vous que la politique est au format JSON.
- Principal non valide : assurez-vous que l'`Principal`élément existe dans la politique de ressources et que la valeur est valide. Si l'`Principal`élément de votre politique de ressources Amazon SQS inclut une entité IAM, assurez-vous que l'entité existe avant d'utiliser la politique. Amazon SQS valide la politique de ressources et vérifie l'entité IAM. Si l'entité IAM n'existe pas, vous recevrez un message d'erreur. Pour confirmer les entités IAM, utilisez les `GetUser`API `GetRole`et.

Pour plus d'informations sur la résolution d'une `InvalidAttributeValue` erreur, consultez Comment résoudre l' QueueDoesNotExist erreur lorsque je passe des appels d'API à ma file d'attente Amazon SQS ? dans le guide du centre de AWS connaissances.

ReceiptHandle erreur

Lors d'un appel d'`DeleteMessage`API, l'erreur `ReceiptHandleIsInvalid` `InvalidParameterValue` peut être renvoyée si le descripteur de réception est incorrect ou a expiré.

- ReceiptHandleIsInvalid erreur : si le descripteur du reçu est incorrect, vous recevrez un message d'erreur similaire à cet exemple :

```
An error occurred (ReceiptHandleIsInvalid) when calling the DeleteMessage operation:
 The input receipt handle <YOUR RECEIPT HANDLE> is not a valid receipt handle.
```

- InvalidParameterValue erreur : si le descripteur du reçu est expiré, vous recevrez un message d'erreur similaire à cet exemple :

```
An error occurred (InvalidParameterValue) when calling the DeleteMessage operation:
 Value <YOUR RECEIPT HANDLE> for parameter ReceiptHandle is invalid. Reason: The
 receipt handle has expired.
```

Causes possibles et mesures d'atténuation :

Le descripteur de réception est créé pour chaque message reçu et n'est valide que pendant la période d'expiration de la visibilité. Lorsque le délai de visibilité expire, le message devient visible dans la file d'attente pour les consommateurs. Lorsque vous recevez à nouveau le message du consommateur, vous recevez un nouvel identifiant de réception. Pour éviter les erreurs de traitement de réception incorrectes ou expirées, utilisez le bon identifiant de réception pour supprimer le message pendant le délai de visibilité de la file d'attente Amazon SQS.

Pour plus d'informations sur la résolution `ReceiptHandle` d'une erreur, consultez Comment résoudre les erreurs « » et ReceiptHandle IsInvalid « InvalidParameter Value » lorsque j'utilise l'appel d'API Amazon DeleteMessage SQS ? dans le guide du centre de AWS connaissances.

Résoudre les problèmes liés à la file d'attente des lettres mortes Amazon SQS et au redrive DLQ

Les rubriques suivantes présentent les causes les plus courantes des problèmes liés à Amazon SQS DLQ et DLQ redrive, et expliquent comment les résoudre. Pour plus d'informations, consultez Comment résoudre les problèmes liés au redrive Amazon SQS DLQ ? dans le guide du centre de AWS connaissances.

Rubriques

Problèmes liés au DLQ

Découvrez les problèmes courants liés à la DLQ et découvrez comment les résoudre.

Rubriques

L'affichage des messages avec la console peut entraîner leur envoi dans une file d'attente de lettres mortes

Amazon SQS affiche un message dans la console selon la stratégie de redirection de la file d'attente correspondante. Par conséquent, si vous consultez un message dans la console le nombre de fois spécifié dans la politique de redrive de la file d'attente correspondante, le message est déplacé vers la file d'attente contenant des lettres mortes de la file d'attente correspondante.

Pour régler ce comportement, vous pouvez procéder de l'une des manières suivantes :

- Augmentez le paramètre Maximum Receives pour la stratégie de redirection de la file d'attente correspondante.

- N'affichez pas les messages de la file d'attente correspondante dans la console.

Les métriques **NumberOfMessagesSent** et **NumberOfMessagesReceived** d'une file d'attente de lettres mortes ne correspondent pas

Si vous envoyez un message à une file d'attente de lettres mortes manuellement, il est capturé par la métrique `NumberOfMessagesSent`. Toutefois, si un message est envoyé à une file d'attente de lettres mortes en raison de l'échec d'une tentative de traitement, il n'est pas capturé par cette métrique. Il est donc possible que les valeurs de `NumberOfMessagesSent` et `NumberOfMessagesReceived`soient différentes.

Création et configuration d'un redrive dans une file d'attente contenant des lettres mortes

Le redrive de la file d'attente de lettres mortes nécessite que vous définissiez les autorisations appropriées pour qu'Amazon SQS puisse recevoir des messages de la file d'attente de lettres mortes et envoyer des messages à la file d'attente de destination. Si vous ne disposez pas des autorisations appropriées, la tâche de redynamisation de la file d'attente contenant des lettres mortes peut échouer. Vous pouvez consulter l'état de votre tâche de retransmission des messages pour résoudre les problèmes, puis réessayer.

Gestion des défaillances des messages de file d'attente standard et FIFO

Les files d'attente standard continuent de traiter les messages jusqu'à l'expiration de la période de conservation. Ce traitement continu réduit les risques de blocage de la file d'attente par des messages non consommés. Le fait d'avoir un grand nombre de messages que le consommateur ne parvient pas à supprimer à plusieurs reprises peut augmenter les coûts et alourdir la charge matérielle. Pour réduire les coûts, déplacez les messages ayant échoué vers la file d'attente des lettres mortes.

Les files d'attente standard autorisent également un grand nombre de messages en vol. Si la majorité de vos messages ne peuvent pas être consommés et ne sont pas envoyés dans une file d'attente de lettres mortes, votre taux de traitement des messages peut ralentir. Pour maintenir l'efficacité de votre file d'attente, assurez-vous que votre application gère correctement le traitement des messages.

Les files d'attente FIFO garantissent un traitement unique en consommant les messages d'un groupe de messages dans l'ordre. Par conséquent, bien que le consommateur puisse continuer à récupérer les messages commandés dans un autre groupe de messages, le premier groupe de messages reste

indisponible tant que le message bloquant la file d'attente n'est pas traité avec succès ou n'est pas déplacé vers une file d'attente de lettres mortes.

De plus, les files d'attente FIFO permettent de réduire le nombre de messages en vol. Pour éviter que votre file d'attente FIFO ne soit bloquée par un message, assurez-vous que votre application gère correctement le traitement des messages.

Pour plus d'informations, consultez Quotas de messages Amazon SQS et Utilisation des messages Amazon SQS.

Problèmes liés au DLQ-Redrive

Découvrez les problèmes courants liés au DLQ-Redrive et découvrez comment les résoudre.

Rubriques

- AccessDenied problème d'autorisation
- NonExistentQueue erreur
- CouldNotDetermineMessageErreur de source

AccessDenied problème d'autorisation

L'`AccessDenied`erreur se produit lorsque le redrive DLQ échoue parce que l'entité AWS Identity and Access Management (IAM) ne dispose pas des autorisations requises.

Exemple de message d'erreur :

```
Failed to create redrive task. Error code: AccessDenied - Queue Permissions to Redrive.
```

Les autorisations d'API suivantes sont requises pour effectuer des demandes de redrive DLQ :

Pour démarrer le redrive d'un message, procédez comme suit :

- Autorisations relatives à la file d'attente des lettres mortes :
  - `sqs:StartMessageMoveTask`
  - `sqs:ReceiveMessage`
  - `sqs:DeleteMessage`
  - `sqs:GetQueueAttributes`

  - `kms:Decrypt`— Lorsque la file d'attente contenant des lettres mortes ou la file source d'origine sont chiffrées.
- Autorisations de file d'attente de destination :
  - `sqs:SendMessage`
  - `kms:GenerateDataKey`— Lorsque la file d'attente de destination est cryptée.
  - `kms:Decrypt` — Lorsque la file d'attente de destination est cryptée.

Pour annuler le renvoi d'un message en cours :

- Autorisations relatives à la file d'attente des lettres mortes :
  - `sqs:CancelMessageMoveTask`
  - `sqs:ReceiveMessage`
  - `sqs:DeleteMessage`
  - `sqs:GetQueueAttributes`
  - `kms:Decrypt`— Lorsque la file d'attente contenant des lettres mortes ou la file source d'origine sont chiffrées.

Pour afficher l'état du déplacement d'un message :

- Autorisations relatives à la file d'attente des lettres mortes :
  - `sqs:ListMessageMoveTasks`
  - `sqs:GetQueueAttributes`

NonExistentQueue erreur

L'`NonExistentQueue`erreur se produit lorsque la file d'attente source Amazon SQS n'existe pas ou a été supprimée. Vérifiez et reconduisez vers une file d'attente Amazon SQS présente.

Exemple de message d'erreur :

```
Failed: AWS.SimpleQueueService.NonExistentQueue
```

CouldNotDetermineMessageErreur de source

L'`CouldNotDetermineMessageSource`erreur se produit lorsque vous tentez de démarrer un redrive DLQ avec les scénarios suivants :

- Un message Amazon SQS envoyé directement au DLQ avec API. `SendMessage`
- Message issu de la rubrique AWS Lambda ou de la fonction Amazon Simple Notification Service (Amazon SNS) avec le DLQ configuré.

Pour résoudre cette erreur, choisissez Redrive vers une destination personnalisée lorsque vous démarrez le redrive. Entrez ensuite l'ARN de la file d'attente Amazon SQS pour déplacer tous les messages de la DLQ vers la file d'attente de destination.

Exemple de message d'erreur :

```
Failed: CouldNotDetermineMessageSource
```

Résoudre les problèmes de régulation FIFO dans Amazon SQS

Par défaut, les files d'attente FIFO prennent en charge 300 transactions par seconde, par action d'API pour `SendMessageReceiveMessage`, et. `DeleteMessage` Les requêtes supérieures à 300 TPS génèrent l'`ThrottlingException`erreur même si les messages de la file d'attente sont disponibles. Pour pallier ce problème, vous pouvez utiliser les méthodes suivantes :

- Activez un débit élevé pour les files d'attente FIFO dans Amazon SQS.
- Utilisez les actions `SendMessageBatch` par lots de l'API Amazon SQS `ChangeMessageVisibilityBatch` pour augmenter la limite TPS de 3 000 messages par seconde et par action d'API, et pour réduire les coûts. `DeleteMessageBatch` Pour l'`ReceiveMessage`API, définissez le `MaxNumberofMessages` paramètre pour recevoir jusqu'à dix messages par transaction. Pour plus d’informations, consultez Actions groupées Amazon SQS.
- Pour les files d'attente FIFO à haut débit, suivez les recommandations pour optimiser l'utilisation des partitions. Envoyez des messages avec les mêmes identifiants de groupe de messages par lots. Supprimez des messages ou modifiez les valeurs du délai d'expiration de visibilité des messages par lots avec des descripteurs de réception provenant des mêmes demandes d'`ReceiveMessage`API.

- Augmentez le nombre de `MessageGroupId`valeurs uniques. Cela permet une distribution uniforme entre les partitions de file d'attente FIFO. Pour plus d'informations, consultez Utilisation de l'ID de groupe de messagerie Amazon SQS.

Pour plus d'informations, consultez Pourquoi ma file d'attente FIFO Amazon SQS ne renvoie pas tous les messages ou les messages des autres groupes de messages ? dans le guide du centre de AWS connaissances.

Résoudre les problèmes liés aux messages non renvoyés lors d'un appel d'API Amazon ReceiveMessage SQS

Les rubriques suivantes décrivent les causes les plus courantes pour lesquelles un message Amazon SQS peut ne pas être renvoyé aux clients, ainsi que la manière de les résoudre. Pour plus d'informations, consultez Pourquoi ne puis-je pas recevoir de messages depuis ma file d'attente Amazon SQS ? dans le guide du centre de AWS connaissances.

Rubriques

File d'attente vide

Pour déterminer si une file d'attente est vide, utilisez une longue interrogation pour appeler l'`ReceiveMessage`API. Vous pouvez également utiliser les `ApproximateNumberOfMessagesDelayed` CloudWatch métriques `ApproximateNumberOfMessagesVisible``ApproximateNumberOfMessagesNotVisible`, et. Si toutes les valeurs métriques sont définies sur 0 pendant plusieurs minutes, la file d'attente est considérée comme vide.

Limite en vol atteinte

Si vous utilisez un long sondage et si la limite de vol de la file d'attente (20 000 pour le FIFO, 120 000 pour le standard par défaut) est dépassée, Amazon SQS ne renverra pas de messages d'erreur dépassant les limites de quota.

Retard du message

Si la file d'attente Amazon SQS est configurée comme une file d'attente différée, ou si les messages ont été envoyés avec des temporisateurs, les messages ne sont pas visibles tant que le délai n'est pas écoulé. Pour vérifier si une file d'attente est configurée en tant que file d'attente différée, utilisez l'`DelaySeconds`attribut `GetQueueAttributes`API ou depuis la console de file d'attente sous Délai de livraison. Vérifiez la `ApproximateNumberOfMessagesDelayed` CloudWatch métrique pour savoir si des messages sont retardés.

Le message est en cours

Si un autre consommateur a interrogé le message, celui-ci sera diffusé ou invisible pendant la période d'expiration du délai de visibilité. Les sondages supplémentaires peuvent renvoyer un reçu vide. Vérifiez la CloudWatch métrique ApproximateNumberOfMessagesVisible pour connaître le nombre de messages pouvant être reçus. Dans le cas des files d'attente FIFO, si un message portant l'ID du groupe de messages est en cours de traitement, aucun autre message ne sera renvoyé, sauf si vous le supprimez ou s'il devient visible. Cela est dû au fait que l'ordre des messages est maintenu au niveau du groupe de messages dans une file d'attente FIFO.

Méthode de sondage

Si vous utilisez un sondage court, (le nombre de WaitTimesecondes est égal à 0) Amazon SQS échantillonne un sous-ensemble de ses serveurs et renvoie des messages provenant uniquement de ces serveurs. Par conséquent, il se peut que vous ne receviez pas les messages même s'ils sont disponibles pour être reçus. Les demandes de sondage suivantes renverront les messages.

Si vous utilisez un long sondage, Amazon SQS interroge tous les serveurs et envoie une réponse après avoir collecté au moins un message disponible, et jusqu'au nombre maximum spécifié. Si la valeur de ReceiveMessage WaitTimeSeconds est trop faible, il est possible que vous ne receviez pas tous les messages disponibles.

Résoudre les erreurs réseau Amazon SQS

Les rubriques suivantes décrivent les causes les plus courantes des problèmes de réseau dans Amazon SQS et expliquent comment les résoudre.

Rubriques

ETIMEOUT error

L'ETIMEOUTerreur se produit lorsque le client ne parvient pas à établir une connexion TCP avec un point de terminaison Amazon SQS.

Résolution de problèmes

- Vérifiez la connexion réseau

  Testez votre connexion réseau à Amazon SQS en exécutant des commandes telles que. `telnet`

  Example: telnet sqs.us-east-1.amazonaws.com 443
- Vérifiez les paramètres réseau
  - Assurez-vous que les règles de pare-feu, les itinéraires et les listes de contrôle d'accès (ACL) locaux autorisent le trafic sur le port que vous utilisez.
  - Les règles de sortie (sortie) du groupe de sécurité doivent autoriser le trafic vers le port 80 ou 443.
  - Les règles de sortie (sortie) ACL du réseau doivent autoriser le trafic vers le port TCP 80 ou 443.
  - Les règles d'entrée (ACL) du réseau doivent autoriser le trafic sur les ports TCP 1024 à 65535.
  - Les instances Amazon Elastic Compute Cloud (Amazon EC2) qui se connectent à l'Internet public doivent disposer d'une connexion Internet.
- Points de terminaison Amazon Virtual Private Cloud (Amazon VPC)

  Si vous accédez à Amazon SQS via un point de terminaison Amazon VPC, le groupe de sécurité des points de terminaison doit autoriser le trafic entrant vers le groupe de sécurité du client sur le port 443. L'ACL réseau associée au sous-réseau du point de terminaison VPC doit avoir cette configuration :

- Les règles de sortie (sortie) ACL du réseau doivent autoriser le trafic sur les ports TCP 1024 à 65535 (ports éphémères).
- Les règles d'entrée (ACL) du réseau doivent autoriser le trafic sur le port 443.

En outre, la politique de point de terminaison VPC AWS Identity and Access Management (IAM) d'Amazon SQS doit autoriser l'accès. L'exemple de politique de point de terminaison VPC suivant indique que l'utilisateur IAM *MyUser*est autorisé à envoyer des messages à la file d'attente Amazon SQS. *MyQueue* L'accès aux autres actions, aux utilisateurs IAM et aux ressources Amazon SQS est refusé via le point de terminaison VPC.

```
{
    "Statement": [{
        "Action": ["sqs:SendMessage"],
        "Effect": "Allow",
        "Resource": "arn:aws:sqs:us-east-2:123456789012:MyQueue",
        "Principal": {
            "AWS": "arn:aws:iam:123456789012:user/MyUser"
        }
    }]
}
```

UnknownHostException error

L'UnknownHostExceptionerreur se produit lorsque l'adresse IP de l'hôte n'a pas pu être déterminée.

Résolution de problèmes

Utilisez l'nslookuputilitaire pour renvoyer l'adresse IP associée au nom d'hôte :

- Windows and Linux OS

```
nslookup sqs.<region>.amazonaws.com
```

- AWS CLI ou SDK pour les anciens points de terminaison Python :

```
nslookup <region>.queue.amazonaws.com
```

Si vous recevez un résultat infructueux, suivez les instructions de la section Comment fonctionne le DNS et comment résoudre les défaillances partielles ou intermittentes du DNS ? dans le guide du centre de AWS connaissances.

Si vous avez reçu une sortie valide, il s'agit probablement d'un problème au niveau de l'application. Pour résoudre les problèmes au niveau de l'application, essayez les méthodes suivantes :

- Redémarrez votre application.
- Vérifiez que le cache DNS de votre application Java n'est pas défectueux. Si possible, configurez votre application pour qu'elle adhère au TTL DNS. Pour plus d'informations, voir Configuration du TTL de la JVM pour les recherches de noms DNS.

Pour plus d'informations sur la résolution des erreurs réseau, consultez Comment résoudre les erreurs de connexion « ETIMEOUT » et UnknownHost « Exception » d'Amazon SQS ? dans le guide du centre de AWS connaissances.

Dépannage des files d'attente Amazon Simple Queue Service avec AWS X-Ray

AWS X-Ray collecte des données sur les demandes traitées par votre application et vous permet de visualiser et de filtrer les données afin d'identifier les problèmes potentiels et les opportunités d'optimisation. Pour toute demande retracée envoyée à votre application, vous pouvez consulter des informations détaillées sur la demande, la réponse et les appels que votre application fait aux AWS ressources en aval, aux microservices, aux bases de données et aux API Web HTTP.

Pour envoyer des en-têtes de AWS X-Ray trace via Amazon SQS, vous pouvez effectuer l'une des opérations suivantes :

- Utilisez l'en-tête de suivi `X-Amzn-Trace-Id`.
- Utilisez l'attribut de système de message `AWSTraceHeader`.

Pour collecter des données sur les erreurs et la latence, vous devez instrumenter le client `AmazonSQS` à l'aide du kit SDK AWS X-Ray.

Vous pouvez utiliser la AWS X-Ray console pour consulter la carte des connexions entre Amazon SQS et les autres services utilisés par votre application. Vous pouvez également utiliser la console

pour afficher des mesures comme la latence moyenne et les taux de défaillance. Pour plus d'informations, consultez Amazon SQS et AWS X-Ray dans le Guide du développeur AWS X-Ray .

Sécurité dans Amazon SQS

Cette section fournit des informations sur la sécurité, l'authentification et le contrôle d'accès dans Amazon SQS, ainsi que sur le langage de la politique d'accès Amazon SQS.

Rubriques

Protection des données dans Amazon SQS

Le modèle de responsabilité AWS partagée s'applique à la protection des données dans Amazon Simple Queue Service. 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 Amazon SQS ou une autre entreprise à Services AWS l'aide de la console, de l'API ou AWS des AWS CLI SDK. 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.

Les sections suivantes présentent des informations sur la protection des données dans Amazon SQS.

Rubriques

- Chiffrement des données dans Amazon SQS
- Confidentialité du trafic interréseau dans Amazon SQS

Chiffrement des données dans Amazon SQS

La protection des données fait référence au fait de protéger les données pendant leur transit (lorsqu'elles sont transmises en direction ou en provenance d'Amazon SQS) et au repos (lorsqu'elles sont stockées sur des disques dans les centres de données Amazon SQS). Vous pouvez protéger les données en transit à l'aide de la technologie Secure Sockets Layer (SSL) ou du chiffrement côté client. Par défaut, Amazon SQS stocke les messages et les fichiers à l'aide du chiffrement du disque. Vous pouvez protéger les données inactives en demandant à Amazon SQS de chiffrer vos messages

avant de les enregistrer dans le système de fichiers chiffré de ses centres de données. Amazon SQS recommande d'utiliser SSE pour optimiser le chiffrement des données.

Rubriques

- Chiffrement au repos dans Amazon SQS
- Gestion des clés Amazon SQS

Chiffrement au repos dans Amazon SQS

Le chiffrement côté serveur (SSE, Server-Side Encryption) vous permet de transférer des données sensibles dans des files d'attente chiffrées. SSE protège le contenu des messages dans les files d'attente à l'aide de clés de chiffrement gérées par SQS (SSE-SQS) ou de clés gérées dans le (SSE-KMS). AWS Key Management Service Pour plus d'informations sur la gestion de l'ESS à l'aide de AWS Management Console, consultez les rubriques suivantes :

- Configuration de SSE-SQS pour une file d'attente (console)
- Configuration de SSE-KMS pour une file d'attente (console)

Pour plus d'informations sur la gestion de l'ESS à l'aide des `GetQueueAttributes` actions AWS SDK for Java (et des`CreateQueue`, `SetQueueAttributes` et), consultez les exemples suivants :

- Utilisation du chiffrement côté serveur avec les files d'attente Amazon SQS
- Configuration des autorisations KMS pour Services AWS

SSE chiffre les messages une fois reçus par Amazon SQS. Les messages sont stockés sous forme chiffrée et Amazon SQS les déchiffre uniquement lorsqu'ils sont envoyés à un consommateur autorisé.

> **Important**
>
> Toutes les demandes adressées aux files d'attente avec le chiffrement SSE activé doivent utiliser HTTPS et Signature version 4.
> Une file d'attente chiffrée qui utilise la clé par défaut (clé KMS AWS gérée pour Amazon SQS) ne peut pas appeler de fonction Lambda dans un autre. Compte AWS

Certaines fonctionnalités des AWS services qui peuvent envoyer des notifications à Amazon SQS à l'aide de cette AWS Security Token Service `AssumeRole` action sont compatibles avec SSE mais ne fonctionnent qu'avec les files d'attente standard :

- Hooks de cycle de vie autoscaling
- AWS Lambda Files d'attente de lettres mortes

Pour plus d'informations sur la compatibilité d'autres services avec les files d'attente chiffrées, consultez Configuration des autorisations KMS pour les AWS services et la documentation de votre service.

AWS KMS combine du matériel et des logiciels sécurisés et hautement disponibles pour fournir un système de gestion des clés adapté au cloud. Lorsque vous utilisez Amazon SQS avec AWS KMS, les clés de données qui chiffrent les données de vos messages sont également chiffrées et stockées avec les données qu'elles protègent.

L'utilisation d' AWS KMS offre les avantages suivants :

- Vous pouvez créer et gérer des AWS KMS keys vous-même.
- Vous pouvez également utiliser la clé KMS AWS gérée pour Amazon SQS, qui est unique pour chaque compte et chaque région.
- Les normes AWS KMS de sécurité peuvent vous aider à respecter les exigences de conformité liées au chiffrement.

Pour plus d'informations, veuillez consulter Présentation de AWS Key Management Service dans le Manuel du développeur AWS Key Management Service .

Rubriques

- Portée du chiffrement
- Termes clés

Portée du chiffrement

Le chiffrement SSE chiffre le corps d'un message dans une file d'attente Amazon SQS.

Le chiffrement SSE ne chiffre pas les éléments suivants :

- métadonnées de la file d'attente (nom et attributs)
- métadonnées du message (ID de message, horodatage et attributs)
- métriques par file d'attente

Le chiffrement d'un message rend son contenu indisponible pour les utilisateurs non autorisés ou anonymes. Lorsque SSE est activé, les demandes anonymes `SendMessage` et `ReceiveMessage` adressées à la file d'attente chiffrée sont rejetées. Les bonnes pratiques de sécurité d'Amazon SQS déconseillent l'utilisation de demandes anonymes. Si vous souhaitez envoyer des demandes anonymes à une file d'attente Amazon SQS, assurez-vous de désactiver SSE. Cela n'affecte pas le fonctionnement normal d'Amazon SQS :

- Un message est chiffré uniquement s'il est envoyé après l'activation du chiffrement d'une file d'attente. Amazon SQS ne chiffre pas les messages en backlog.
- Tout message chiffré reste chiffré même si le chiffrement de sa file d'attente est désactivé.

Le placement d'un message dans une file d'attente de lettres mortes n'affecte pas son chiffrement :

- Lorsqu'Amazon SQS transfère un message d'une file d'attente source chiffrée vers une file d'attente de lettres mortes non chiffrée, le message reste chiffré.
- Lorsqu'Amazon SQS transfère un message d'une file d'attente source non chiffrée vers une file d'attente de lettres mortes chiffrée, le message reste non chiffré.

Termes clés

Les termes clés suivants peuvent vous aider à mieux comprendre les fonctionnalités de chiffrement SSE. Pour des descriptions détaillées, consultez la Référence de l'API Amazon Simple Queue Service.

Clé de données

Clé (DEK) permettant de chiffrer le contenu des messages Amazon SQS.

Pour plus d'informations, consultez la section Clés de données du Guide du développeur AWS Key Management Service et du Guide du développeur AWS Encryption SDK .

Période de réutilisation des clés de données

Durée, en secondes, pendant laquelle Amazon SQS peut réutiliser une clé de données pour chiffrer ou déchiffrer des messages avant de réappeler. AWS KMS Entier en secondes, compris entre 60 secondes (1 minute) et 86 400 secondes (24 heures). La valeur par défaut est 300 (5 minutes). Pour plus d'informations, consultez Présentation de la période de réutilisation des clés de données.

> **Note**
>
> Dans le cas peu probable où il serait impossible de l'atteindre AWS KMS, Amazon SQS continue d'utiliser la clé de données mise en cache jusqu'à ce qu'une connexion soit rétablie.

ID de clé KMS

Alias, alias ARN, ID de clé ou ARN de clé d'une clé KMS AWS gérée ou d'une clé KMS personnalisée, dans votre compte ou dans un autre compte. Bien que l'alias de la clé KMS AWS gérée pour Amazon SQS soit toujours`alias/aws/sqs`, l'alias d'une clé KMS personnalisée peut, par exemple, être. `alias/MyAlias` Vous pouvez utiliser ces clés KMS pour protéger les messages des files d'attente Amazon SQS.

> **Note**
>
> Gardez à l'esprit les points suivants :
>
> - Si vous ne spécifiez pas de clé KMS personnalisée, Amazon SQS utilise la clé KMS AWS gérée pour Amazon SQS.
> - La première fois que vous utilisez le AWS Management Console pour spécifier la clé KMS AWS gérée pour Amazon SQS pour une file d'attente, la clé KMS AWS gérée est AWS KMS créée pour Amazon SQS.
> - Sinon, la première fois que vous utilisez l'`SendMessageBatch`action `SendMessage` ou sur une file d'attente avec SSE activé, vous AWS KMS créez la clé KMS AWS gérée pour Amazon SQS.

Vous pouvez créer des clés KMS, définir les politiques qui contrôlent la manière dont les clés KMS peuvent être utilisées et auditer l'utilisation des clés KMS à l'aide de la section Clés

gérées par le client de la AWS KMS console ou de l'`CreateKey` AWS KMS action. Pour plus d'informations, consultez Clés KMS et Création de clés du Guide du développeur AWS Key Management Service . Pour plus d'exemples d'identifiants de clé KMS, consultez KeyIdla référence AWS Key Management Service d'API. Pour plus d'informations sur la recherche d'identifiants de clés KMS, consultez la section Recherche de l'ID et de l'ARN d'une clé du Guide du développeur AWS Key Management Service .

> **Important**
>
> L'utilisation entraîne des frais supplémentaires AWS KMS. Pour plus d'informations, veuillez consulter les sections Estimation AWS KMS des coûts et Tarification d'AWS Key Management Service.

Chiffrement d'enveloppe

La sécurité de vos données chiffrées dépend partiellement de la protection de la clé de données capable de les déchiffrer. Amazon SQS utilise la clé KMS pour chiffrer la clé de données, puis la clé de données chiffrée est stockée avec le message chiffré. La pratique qui consiste à utiliser une clé KMS pour chiffrer des clés de données s'appelle le chiffrement d'enveloppe.

Pour plus d'informations, consultez Chiffrement d'enveloppe dans le Guide du développeur AWS Encryption SDK .

Gestion des clés Amazon SQS

Amazon SQS s'intègre au AWS Key Management Service (KMS) pour gérer les clés KMS pour le chiffrement côté serveur (SSE). Consultez Chiffrement au repos dans Amazon SQS pour obtenir des informations sur le SSE et les définitions de la gestion des clés. Amazon SQS utilise des clés KMS pour valider et sécuriser les clés de données qui chiffrent et déchiffrent les messages. Les sections suivantes fournissent des informations sur l'utilisation des clés KMS et de données dans le service Amazon SQS.

Rubriques

- Configuration des autorisations AWS KMS
- Présentation de la période de réutilisation des clés de données
- Estimation AWS KMS des coûts

- AWS KMS erreurs

Configuration des autorisations AWS KMS

Chaque clé KMS doit avoir une politique de clé. Notez que vous ne pouvez pas modifier la politique de clé d'une clé KMS AWS gérée pour Amazon SQS. La stratégie de cette clé KMS inclut des autorisations permettant à tous les mandataires du compte (autorisés à utiliser Amazon SQS) d'utiliser des files d'attente chiffrées.

Pour une clé Amazon SQS gérée par le client, vous devez configurer la stratégie de clé afin d'ajouter des autorisations pour chaque producteur et consommateur de file d'attente. Pour ce faire, vous nommez le producteur et le consommateur en tant qu'utilisateurs dans la stratégie de clé KMS. Pour plus d'informations sur AWS KMS les autorisations, consultez les AWS KMS ressources et les opérations ou la référence aux autorisations d'AWS KMS API dans le guide du AWS Key Management Service développeur.

Vous pouvez également spécifier les autorisations requises dans une stratégie IAM affectée aux mandataires qui produisent et consomment des messages chiffrés. Pour de plus amples informations, veuillez consulter Utilisation des stratégies IAM avec AWS KMS dans le Guide du développeur AWS Key Management Service .

> **Note**
>
> Bien que vous puissiez configurer des autorisations globales pour envoyer et recevoir depuis Amazon SQS, vous devez nommer AWS KMS explicitement l'ARN complet des clés KMS dans des régions spécifiques dans la `Resource` section d'une politique IAM.

Configuration des autorisations KMS pour les AWS services

Plusieurs AWS services agissent comme des sources d'événements qui peuvent envoyer des événements aux files d'attente Amazon SQS. Pour permettre à ces sources d'événements de fonctionner avec des files d'attente chiffrées, vous devez créer une clé KMS gérée par le client et ajouter des autorisations dans la politique des clés afin que le service utilise les méthodes d' AWS KMS API requises. Effectuez les étapes suivantes pour configurer les autorisations.

> **Warning**
>
> Lorsque vous modifiez la clé KMS pour chiffrer vos messages Amazon SQS, sachez que les messages existants chiffrés avec l'ancienne clé KMS resteront chiffrés avec cette clé. Pour déchiffrer ces messages, vous devez conserver l'ancienne clé KMS et vous assurer que sa politique en matière de clés accorde à Amazon SQS les autorisations `kms:Decrypt` pour et. `kms:GenerateDataKey` Après la mise à jour vers une nouvelle clé KMS pour chiffrer les nouveaux messages, assurez-vous que tous les messages existants chiffrés avec l'ancienne clé KMS sont traités et retirés de la file d'attente avant de supprimer ou de désactiver l'ancienne clé KMS.

1. Créez une clé KMS gérée par le client. Pour plus d'informations, consultez Création des clés dans le Guide du développeur AWS Key Management Service .
2. Pour autoriser la source de l'événement de AWS service à utiliser les méthodes `kms:GenerateDataKey` et `kms:Decrypt` API, ajoutez l'instruction suivante à la politique de clé KMS.

```
{
   "Version": "2012-10-17",
      "Statement": [{
         "Effect": "Allow",
         "Principal": {
            "Service": "service.amazonaws.com"
         },
         "Action": [
            "kms:GenerateDataKey",
            "kms:Decrypt"
         ],
         "Resource": "*"
       }]
}
```

Remplacez « service » dans l'exemple ci-dessus par le nom de service de la source de l'événement. Les sources d'événements incluent les services suivants.

Source de l'événement	Nom du service
CloudWatch Événements Amazon	`events.amazonaws.com`
Notifications d'événements Amazon S3	`s3.amazonaws.com`
Abonnements à des rubriques Amazon SNS	`sns.amazonaws.com`

3. Configurez une file d'attente SSE existante à l'aide de l'ARN de votre clé KMS.
4. Fournissez l'ARN de la file d'attente chiffrée à la source de l'événement.

Configurer AWS KMS les autorisations pour les producteurs

Lorsque la période de réutilisation de la clé de données expire, le prochain appel du producteur vers `SendMessage` ou `SendMessageBatch` déclenche également des appels vers `kms:GenerateDataKey` et `kms:Decrypt`. L'appel à `kms:Decrypt` a pour but de vérifier l'intégrité de la nouvelle clé de données avant de l'utiliser. Par conséquent, le producteur doit disposer des autorisations `kms:GenerateDataKey` et `kms:Decrypt` pour la clé KMS.

Ajoutez l'instruction suivante à la stratégie IAM du producteur. N'oubliez pas d'utiliser les valeurs ARN correctes pour la ressource clé et la ressource de file d'attente.

```
{
   "Version": "2012-10-17",
      "Statement": [{
         "Effect": "Allow",
         "Action": [
            "kms:GenerateDataKey",
            "kms:Decrypt"
         ],
         "Resource":  "arn:aws:kms:us-
east-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
         }, {
         "Effect": "Allow",
         "Action": [
            "sqs:SendMessage"
         ],
         "Resource": "arn:aws:sqs:*:123456789012:MyQueue"
      }]
```

```
}
```

Configurer AWS KMS les autorisations pour les consommateurs

Lorsque la période de réutilisation de la clé de données expire, le prochain appel du consommateur vers `ReceiveMessage` déclenche également un appel de `kms:Decrypt`, pour vérifier l'intégrité de la nouvelle clé de données avant de l'utiliser. Par conséquent, le consommateur doit disposer de l'autorisation `kms:Decrypt` pour toute clé KMS permettant de chiffrer les messages dans la file d'attente spécifiée. Si la file d'attente sert de file d'attente de lettres mortes, le consommateur doit également disposer de l'autorisation `kms:Decrypt` pour toute clé KMS permettant de chiffrer les messages dans la file d'attente source. Ajoutez l'instruction suivante à la stratégie IAM du consommateur. N'oubliez pas d'utiliser les valeurs ARN correctes pour la ressource clé et la ressource de file d'attente.

```
{
   "Version": "2012-10-17",
      "Statement": [{
         "Effect": "Allow",
         "Action": [
            "kms:Decrypt"
         ],
         "Resource": "arn:aws:kms:us-
east-2:123456789012:key/1234abcd-12ab-34cd-56ef-1234567890ab"
         }, {
         "Effect": "Allow",
         "Action": [
            "sqs:ReceiveMessage"
         ],
         "Resource": "arn:aws:sqs:*:123456789012:MyQueue"
      }]
}
```

Configurer AWS KMS les autorisations avec une protection adjointe confuse

Lorsque le mandataire dans une instruction de stratégie de clé est un principal de service AWS, vous pouvez utiliser les clés de condition globales `aws:SourceArn` ou `aws:SourceAccount` pour vous protéger contre le scénario de député confus. Pour utiliser ces clés de condition, définissez comme valeur l'Amazon Resource Name (ARN) de la ressource à chiffrer. Si vous ne connaissez pas l'ARN de la ressource, utilisez `aws:SourceAccount` à la place.

Dans cette stratégie de clé KMS, une ressource spécifique issue d'un service appartenant au compte `111122223333` est autorisée à appeler KMS pour les actions `Decrypt` et `GenerateDataKey`, qui se produisent lors de l'utilisation du SSE d'Amazon SQS.

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Principal": {
            "Service": "<replaceable>service</replaceable>.amazonaws.com"
  },
  "Action": [
   "kms:GenerateDataKey",
   "kms:Decrypt"
  ],
  "Resource": "*",
  "Condition": {
   "ArnEquals": {
    "aws:SourceArn": [
     "arn:aws:service::111122223333:resource"
    ]
   }
  }
 }]
}
```

Lorsque vous utilisez des files d'attente Amazon SQS compatibles SSE, les services suivants prennent en charge `aws:SourceArn` :

- Amazon SNS
- Amazon S3
- CloudWatch Évènements
- AWS Lambda
- CodeBuild
- Profils des clients Amazon Connect
- AWS Auto Scaling
- Amazon Chime

Présentation de la période de réutilisation des clés de données

La période de réutilisation des clés de données définit la durée maximale de réutilisation de la même clé de données par Amazon SQS. Lorsque la période de réutilisation de la clé de données se termine, Amazon SQS génère une nouvelle clé de données. Notez les instructions suivantes concernant la période de réutilisation.

- Une période de réutilisation plus courte améliore la sécurité, mais entraîne un plus grand nombre d'appels AWS KMS, ce qui peut entraîner des frais au-delà du niveau gratuit.
- Bien que la clé de données soit mise en cache séparé pour le chiffrement et le déchiffrement, la période de réutilisation s'applique aux deux copies de cette clé.
- Lorsque la période de réutilisation des clés de données prend fin, le prochain appel `SendMessage` ou déclenche `SendMessageBatch` généralement un appel à la AWS KMS `GenerateDataKey` méthode pour obtenir une nouvelle clé de données. De plus, les prochains appels à `SendMessage` et `ReceiveMessage` déclencheront chacun un appel AWS KMS `Decrypt` pour vérifier l'intégrité de la clé de données avant de l'utiliser.
- Les principaux (Comptes AWS ou utilisateurs) ne partagent pas les clés de données (les messages envoyés par des principaux uniques reçoivent toujours des clés de données uniques). Par conséquent, le volume d'appels AWS KMS est un multiple du nombre de principaux uniques utilisés pendant la période de réutilisation des clés de données.

Estimation AWS KMS des coûts

Pour prévoir les coûts et mieux comprendre votre AWS facture, vous souhaiterez peut-être savoir à quelle fréquence Amazon SQS utilise votre clé KMS.

> **Note**
>
> Bien que la formule ci-après puisse vous donner une très bonne idée des coûts à prévoir, les coûts réels risquent d'être plus élevés en raison de la nature distribuée d'Amazon SQS.

Pour calculer le nombre de demandes d'API (R) par file d'attente, utilisez la formule suivante :

```
R = (B / D) * (2 * P + C)
```

B est la période de facturation (en secondes).

D est la période de réutilisation des clés de données (en secondes).

P est le nombre de mandataires productifs qui envoient des messages à la file d'attente Amazon SQS.

C est le nombre de mandataires consommateurs qui reçoivent des messages de la file d'attente Amazon SQS.

> **Important**
>
> En général, les mandataires productifs ont un coût deux fois plus élevé que celui des mandataires consommateurs. Pour plus d'informations, consultez Présentation de la période de réutilisation des clés de données.
> Si le producteur et le consommateur ont des utilisateurs différents, le coût augmente.

Voici des exemples de calcul. Pour obtenir des informations précises sur la tarification, consultez Tarification AWS Key Management Service.

Exemple 1 : calcul du nombre d'appels d' AWS KMS API pour 2 principaux et 1 file d'attente

Cet exemple suppose que :

- La période de facturation va du 1er au 31 janvier (2 678 400 secondes).
- La période de réutilisation des clés de données est définie sur 5 minutes (300 secondes).
- Il y a 1 file d'attente.
- Il y a 1 mandataire productif et 1 mandataire consommateur.

```
(2,678,400 / 300) * (2 * 1 + 1) = 26,784
```

Exemple 2 : calcul du nombre d'appels d' AWS KMS API pour plusieurs producteurs et consommateurs et pour deux files d'attente

Cet exemple suppose que :

- La période de facturation va du 1er au 28 février (2 419 200 secondes).
- La période de réutilisation des clés de données est définie sur 24 heures (86 400 secondes).
- Il y a 2 files d'attente.

- La première file d'attente comporte 3 mandataires productifs et 1 mandataire consommateur.
- La seconde file d'attente comporte 5 mandataires productifs et 2 mandataires consommateurs.

```
(2,419,200 / 86,400 * (2 * 3 + 1)) + (2,419,200 / 86,400 * (2 * 5 + 2)) = 532
```

AWS KMS erreurs

Lorsque vous travaillez avec Amazon SQS AWS KMS, vous pouvez rencontrer des erreurs. Les références suivantes décrivent les erreurs et les solutions de dépannage possibles.

- Erreurs AWS KMS courantes
- Erreurs de déchiffrement AWS KMS
- AWS KMS GenerateDataKey erreurs

Confidentialité du trafic interréseau dans Amazon SQS

Un point de terminaison d'Amazon Virtual Private Cloud (Amazon VPC) pour Amazon SQS est une entité logique au sein d'un VPC qui autorise la connectivité uniquement à Amazon SQS. Le VPC achemine les demandes vers Amazon SQS et les réponses en retour vers le VPC. Les sections suivantes fournissent des informations sur l'utilisation des points de terminaison de VPC et la création de politiques de point de terminaison de VPC.

Rubriques

- Points de terminaison Amazon Virtual Private Cloud pour Amazon SQS
- Création d'une stratégie de point de terminaison d'un VPC Amazon pour Amazon SQS

Points de terminaison Amazon Virtual Private Cloud pour Amazon SQS

Si vous utilisez Amazon VPC pour héberger vos AWS ressources, vous pouvez établir une connexion entre votre VPC et Amazon SQS. Vous pouvez utiliser cette connexion pour envoyer des messages à vos files d'attente Amazon SQS sans passer par l'Internet public.

Amazon VPC vous permet de lancer AWS des ressources dans un réseau virtuel personnalisé. Vous pouvez utiliser un VPC pour contrôler vos paramètres réseau, tels que la plage d'adresses IP, les sous-réseaux, les tables de routage et les passerelles réseau. Pour plus d'informations sur les VPC, consultez le Guide de l'utilisateur Amazon VPC.

Pour connecter votre VPC à Amazon SQS, vous devez commencer par définir le point de terminaison d'un VPC d'interface, qui vous permet de connecter votre VPC à d'autres services AWS . Le point de terminaison assure une connectivité évolutive et fiable à Amazon SQS, sans qu'une passerelle Internet, une instance de traduction d'adresses réseau (NAT) ou une connexion VPN ne soit nécessaire. Pour plus d'informations, consultez les sections Didacticiel : envoi d'un message à une file d'attente Amazon SQS à partir d'Amazon Virtual Private Cloud et Exemple 5 : Refuser l'accès s'il n'émane pas d'un point de terminaison de VPC de ce guide et la section Points de terminaison d'un VPC d'interface (AWS PrivateLink) du Guide de l'utilisateur Amazon VPC.

Important

- Vous pouvez utiliser Amazon Virtual Private Cloud uniquement avec des points de terminaison HTTPS Amazon SQS.
- Lorsque vous configurez Amazon SQS pour que les messages soient envoyés depuis Amazon VPC, vous devez activer le DNS privé et spécifier les points de terminaison au format `sqs.us-east-2.amazonaws.com`.
- Le DNS privé ne prend pas en charge les points de terminaison existants, tels que `queue.amazonaws.com` ou `us-east-2.queue.amazonaws.com`.

Création d'une stratégie de point de terminaison d'un VPC Amazon pour Amazon SQS

Vous pouvez créer une stratégie pour les points de terminaison d'un VPC Amazon pour Amazon SQS dans laquelle vous spécifiez les éléments suivants :

- Le principal qui peut exécuter des actions.
- Les actions qui peuvent être effectuées.
- Les ressources sur lesquelles les actions peuvent être exécutées.

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.

L'exemple suivant de stratégie de point de terminaison d'un VPC spécifie que l'utilisateur `MyUser` est autorisé à envoyer des messages dans la file d'attente Amazon SQS `MyQueue`.

```
{
   "Statement": [{
```

```
      "Action": ["sqs:SendMessage"],
      "Effect": "Allow",
      "Resource": "arn:aws:sqs:us-east-2:123456789012:MyQueue",
      "Principal": {
        "AWS": "arn:aws:iam:123456789012:user/MyUser"
      }
   }]
}
```

Ce qui suit est refusé :

- Autres actions d'API Amazon SQS, telles que `sqs:CreateQueue` et `sqs:DeleteQueue`.
- Autres utilisateurs et règles qui tentent d'utiliser ce point de terminaison de VPC.
- Envoi de messages par `MyUser` à une autre file d'attente Amazon SQS.

> **Note**
>
> L'utilisateur peut toujours utiliser d'autres actions d'API Amazon SQS depuis l'extérieur du VPC. Pour plus d'informations, voir Exemple 5 : Refuser l'accès s'il n'émane pas d'un point de terminaison de VPC.

Gestion des identités et des accès dans Amazon SQS

AWS Identity and Access Management (IAM) est un outil Service AWS qui permet à un administrateur de contrôler en toute sécurité l'accès aux AWS ressources. Des administrateurs IAM contrôlent les personnes qui peuvent être authentifiées (connectées) et autorisées (disposant d'autorisations) à utiliser des ressources Amazon SQS. IAM est un Service AWS outil que vous pouvez utiliser sans frais supplémentaires.

Public ciblé

La façon dont vous utilisez AWS Identity and Access Management (IAM) varie en fonction du travail que vous effectuez dans Amazon SQS.

Utilisateur du service : si vous utilisez le service Amazon SQS pour accomplir votre tâche, votre administrateur vous fournira les informations d'identification et les autorisations nécessaires. Vous pourrez avoir besoin d'autorisations supplémentaires si vous utilisez davantage de fonctionnalités Amazon SQS. En comprenant bien la gestion des accès, vous saurez demander les autorisations

appropriées à votre administrateur. Si vous ne pouvez pas accéder à une fonctionnalité dans Amazon SQS, consultez Résolution des problèmes d'accès et d'identité Amazon Simple Queue Service.

Administrateur du service : si vous êtes le responsable des ressources Amazon SQS de votre entreprise, vous bénéficiez probablement d'un accès total à ce service. C'est à vous de déterminer les fonctions et les ressources Amazon SQS auxquelles vos utilisateurs des services pourront 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. Pour découvrir la façon dont votre entreprise peut utiliser IAM avec Amazon SQS, veuillez consulter Fonctionnement d'Amazon Simple Queue Service avec IAM.

Administrateur IAM : si vous êtes un administrateur IAM, vous pouvez souhaiter obtenir des informations sur l'écriture de stratégies pour gérer l'accès à Amazon SQS. Pour afficher des exemples de stratégies basées sur l'identité Amazon SQS que vous pouvez utiliser dans IAM, consultez Bonnes pratiques en matière de politiques.

Authentification par des identités

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 la section 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.

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. 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, consultez Tâches nécessitant des informations d'identification d'utilisateur root dans le Guide de l'utilisateur IAM.

Identité fédérée

La meilleure pratique consiste à obliger les utilisateurs humains, y compris ceux qui ont besoin d'un accès administrateur, à utiliser la fédération avec un fournisseur d'identité pour accéder à l'aide Services AWS d'informations d'identification temporaires.

Une identité fédérée est un utilisateur de l'annuaire des utilisateurs de votre entreprise, d'un fournisseur d'identité Web AWS Directory Service, du répertoire Identity Center ou de tout utilisateur qui y accède à l'aide des informations d'identification fournies Services AWS par le biais d'une source d'identité. Lorsque des identités fédérées y accèdent Comptes AWS, elles assument des rôles, qui fournissent des informations d'identification temporaires.

Pour une gestion des accès centralisée, nous vous recommandons d'utiliser AWS IAM Identity Center. Vous pouvez créer des utilisateurs et des groupes dans IAM Identity Center, ou vous pouvez vous connecter et synchroniser avec un ensemble d'utilisateurs et de groupes dans votre propre

source d'identité afin de les utiliser dans toutes vos applications Comptes AWS et applications. Pour obtenir des informations sur IAM Identity Center, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Utilisateurs et groupes IAM

Un utilisateur IAM est une identité au sein de vous Compte AWS qui possède des autorisations spécifiques pour une seule personne ou une seule 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.

Rôles 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 connaître la différence entre les rôles et les politiques basées sur les ressources pour l'accès entre comptes, consultez la section Accès aux ressources entre comptes dans IAM 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.

Gestion des accès à l'aide de politiques

Vous contrôlez l'accès en AWS créant des politiques et en les associant à AWS des identités ou à des 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 (utilisateur, utilisateur root ou session de 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. Pour plus d'informations sur la structure et le contenu des documents de politique JSON, consultez Vue d'ensemble des politiques JSON dans le Guide de l'utilisateur IAM.

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

Par défaut, les utilisateurs et les rôles ne disposent d'aucune autorisation. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM. L'administrateur peut ensuite ajouter les politiques IAM aux rôles et les utilisateurs peuvent assumer les rôles.

Les politiques IAM définissent les autorisations d'une action, quelle que soit la méthode que vous utilisez pour exécuter l'opération. Par exemple, supposons que vous disposiez d'une politique qui autorise l'action `iam:GetRole`. Un utilisateur appliquant cette politique peut obtenir des informations sur le rôle à partir de AWS Management Console AWS CLI, de ou de l' AWS API.

Politiques basées sur l'identité

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Création de politiques IAM dans le Guide de l'utilisateur IAM.

Les politiques basées sur l'identité peuvent être classées comme des politiques en ligne ou des politiques gérées. Les politiques en ligne sont intégrées directement à un utilisateur, groupe ou rôle. Les politiques gérées sont des politiques autonomes que vous pouvez associer à plusieurs utilisateurs, groupes et rôles au sein de votre Compte AWS. Les politiques gérées incluent les politiques AWS gérées et les politiques gérées par le client. Pour découvrir comment choisir entre une politique gérée et une politique en ligne, consultez Choix entre les politiques gérées et les politiques en ligne dans le Guide de l'utilisateur IAM.

politiques basées sur les ressources

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Des politiques basées sur les ressources sont, par exemple, les politiques de confiance de rôle IAM et des politiques de compartiment. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS

Les politiques basées sur les ressources sont des politiques en ligne situées dans ce service. Vous ne pouvez pas utiliser les politiques AWS gérées par IAM dans une stratégie basée sur les ressources.

Listes de contrôle d’accès (ACL)

Les listes de contrôle d’accès (ACL) vérifie quels principals (membres de compte, utilisateurs ou rôles) ont l’autorisation d’accéder à une ressource. Les listes de contrôle d’accès sont similaires aux politiques basées sur les ressources, bien qu’elles n’utilisent 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, consultez Vue d’ensemble des listes de contrôle d’accès (ACL) dans le Guide du développeur Amazon Simple Storage Service.

Autres types de politique

AWS prend en charge d'autres types de politiques moins courants. Ces types de politiques peuvent définir le nombre maximum d’autorisations qui vous sont accordées par des types de politiques plus courants.

- Limite d’autorisations : une limite d’autorisations est une fonctionnalité avancée dans laquelle vous définissez le nombre maximal d’autorisations qu’une politique basée sur l’identité peut accorder à une entité IAM (utilisateur ou rôle IAM). Vous pouvez définir une limite d’autorisations pour une entité. Les autorisations en résultant représentent la combinaison des politiques basées sur l’identité d’une entité et de ses limites d’autorisation. Les politiques basées sur les ressources qui spécifient l’utilisateur ou le rôle dans le champ `Principal` ne sont pas limitées par les limites d’autorisations. Un refus explicite dans l’une de ces politiques remplace l’autorisation. Pour plus d’informations sur les limites d’autorisations, consultez Limites d’autorisations pour des entités IAM dans le Guide de l’utilisateur IAM.
- Politiques de contrôle des services (SCP) — Les SCP sont des politiques JSON qui spécifient les autorisations maximales pour une organisation ou une unité organisationnelle (UO) dans. AWS Organizations AWS Organizations est un service permettant de regrouper et de gérer de manière centralisée Comptes AWS les multiples propriétés de 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. 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. Pour plus d’informations sur les organisations et les SCP, consultez Fonctionnement des SCP dans le Guide de l’utilisateur AWS Organizations .
- Politiques de séance : les politiques de séance sont des politiques avancées que vous utilisez en tant que paramètre lorsque vous créez par programmation une séance temporaire pour un rôle ou un utilisateur fédéré. Les autorisations de séance en résultant sont une combinaison des politiques

basées sur l'identité de l'utilisateur ou du rôle et des politiques de séance. Les autorisations peuvent également provenir d'une politique basée sur les ressources. Un refus explicite dans l'une de ces politiques annule l'autorisation. Pour plus d'informations, consultez politiques de séance dans le Guide de l'utilisateur IAM.

Plusieurs types de politique

Lorsque plusieurs types de politiques s'appliquent à la requête, les autorisations en résultant sont plus compliquées à comprendre. Pour savoir comment AWS déterminer s'il faut autoriser une demande lorsque plusieurs types de politiques sont impliqués, consultez la section Logique d'évaluation des politiques dans le guide de l'utilisateur IAM.

Présentation de la gestion de l'accès dans Amazon SQS

Chaque AWS ressource appartient à un Compte AWS, et les autorisations de création ou d'accès à une ressource sont régies par des politiques d'autorisation. Un administrateur de compte peut accorder des stratégies d'autorisation à des identités IAM (utilisateurs, groupes et rôles), et certains services (tels qu'Amazon SQS) prennent également en charge l'octroi de stratégies d'autorisation à des ressources.

> **Note**
>
> Un administrateur de compte (ou utilisateur administrateur) est un utilisateur doté des privilèges d'administration. Pour plus d'informations, consultez Bonnes pratiques IAM dans le Guide de l'utilisateur IAM.

Lorsque vous accordez des autorisations, vous indiquez quels utilisateurs en bénéficient, à quelles ressources ces autorisations s'appliquent et les actions spécifiques que vous souhaitez autoriser sur la ressource.

Rubriques

- Ressources et opérations Amazon Simple Queue Service
- Présentation de la propriété des ressources
- Gestion de l'accès aux ressources
- Spécification des éléments d'une politique : actions, effets, ressources et principaux

Ressources et opérations Amazon Simple Queue Service

Dans Amazon SQS, la file d'attente est la seule ressource. Dans une stratégie, utilisez un Amazon Resource Name (ARN) pour identifier la ressource à laquelle la stratégie s'applique. La ressource suivante est dotée d'un ARN unique qui lui est associé :

Type de ressource	Format ARN
File d'attente	`arn:aws:sqs:` *`region`*`:`*`account_id`*`:`*`queue_name`*

Vous trouverez ci-dessous des exemples de format ARN pour les files d'attente :

- Un ARN pour une file d'attente nommée `my_queue` dans la région USA Est (Ohio), appartenant au AWS compte 123456789012 :

  ```
  arn:aws:sqs:us-east-2:123456789012:my_queue
  ```

- ARN d'une file d'attente nommée `my_queue` dans chacune des différentes régions prises en charge par Amazon SQS :

  ```
  arn:aws:sqs:*:123456789012:my_queue
  ```

- ARN utilisant `*` ou `?` comme caractère générique pour le nom de la file d'attente. Dans les exemples suivants, l'ARN correspond à toutes les files d'attente dont le préfixe est `my_prefix_` :

  ```
  arn:aws:sqs:*:123456789012:my_prefix_*
  ```

Vous pouvez obtenir la valeur de l'ARN d'une file d'attente existante en appelant l'action `GetQueueAttributes`. La valeur de l'attribut `QueueArn` correspond à l'ARN de la file d'attente. Pour plus d'informations sur les ARN, consultez ARN IAM dans le Guide de l'utilisateur IAM.

Amazon SQS fournit un ensemble d'actions qui fonctionnent avec la ressource de file d'attente. Pour plus d’informations, consultez Autorisations d'API Amazon SQS : référence des actions et ressources.

Présentation de la propriété des ressources

Il Compte AWS est propriétaire des ressources créées dans le compte, quelle que soit la personne qui les a créées. Plus précisément, le propriétaire des ressources est le Compte AWS de l'entité du mandataire (à savoir, le compte racine, un utilisateur ou un rôle IAM) qui authentifie la demande de création des ressources. Les exemples suivants illustrent comment cela fonctionne :

- Si vous utilisez les informations d'identification de votre compte root Compte AWS pour créer une file d'attente Amazon SQS, vous êtes le propriétaire de la ressource (dans Amazon SQS, la ressource Compte AWS est la file d'attente Amazon SQS).
- Si vous créez un utilisateur dans votre file d'attente Compte AWS et que vous lui accordez l'autorisation de créer une file d'attente, celui-ci peut créer la file d'attente. Toutefois, votre Compte AWS (auquel l'utilisateur appartient) est propriétaire de la ressource file d'attente.
- Si vous créez un rôle IAM Compte AWS avec les autorisations nécessaires pour créer une file d'attente Amazon SQS, toute personne capable d'assumer ce rôle peut créer une file d'attente. Vous Compte AWS (à qui appartient le rôle) êtes propriétaire de la ressource de file d'attente.

Gestion de l’accès aux ressources

Une stratégie d'autorisation décrit qui a accès à quoi. La section suivante explique les options disponibles pour créer des stratégies d'autorisation.

> **Note**
>
> Cette section décrit l'utilisation d'IAM dans le contexte d'Amazon SQS. Elle ne fournit pas d'informations détaillées sur le service IAM. Pour une documentation complète sur IAM, consultez la rubrique Qu'est-ce que IAM ? dans le Guide de l'utilisateur IAM. Pour plus d'informations sur la syntaxe et les descriptions des stratégies IAM, consultez Référence de stratégie AWS IAM dans le Guide de l'utilisateur IAM.

Les politiques attachées à une identité IAM sont appelées politiques basées sur une entité (politiques IAM) et les politiques attachées à une ressource sont appelées politiques basées sur une ressource.

Politiques basées sur l’identité

Deux options s'offrent à vous pour autoriser les utilisateurs à accéder à vos files d'attente Amazon SQS : le système de stratégies Amazon SQS ou le système de stratégies IAM. Vous pouvez

utiliser l'un et/ou l'autre des systèmes pour associer des stratégies à des utilisateurs ou à des rôles. Dans la plupart des cas, vous obtenez le même résultat avec l'un ou l'autre système. Par exemple, vous pouvez effectuer les opérations suivantes :

- Attacher une stratégie d'autorisation à un utilisateur ou à un groupe de votre compte : pour autoriser un utilisateur à créer une file d'attente Amazon SQS, attachez une stratégie d'autorisations à cet utilisateur ou à un groupe auquel il appartient.
- Attacher une stratégie d'autorisation à un utilisateur d'un autre compte Compte AWS : pour autoriser un utilisateur à créer une file d'attente Amazon SQS, attachez une stratégie d'autorisations Amazon SQS à cet utilisateur dans un autre Compte AWS.

  Les autorisations intercompte ne s'appliquent pas aux actions suivantes :

  - `AddPermission`
  - `CancelMessageMoveTask`
  - `CreateQueue`
  - `DeleteQueue`
  - `ListMessageMoveTask`
  - `ListQueues`
  - `ListQueueTags`
  - `RemovePermission`
  - `SetQueueAttributes`
  - `StartMessageMoveTask`
  - `TagQueue`
  - `UntagQueue`
- Attacher une stratégie d'autorisation à un rôle (accorder des autorisations intercompte) : pour accorder des autorisations intercompte, attachez une stratégie d'autorisations basée sur une identité à un rôle IAM. Par exemple, l'administrateur Compte AWS A peut créer un rôle pour accorder des autorisations entre comptes à Compte AWS B (ou à un AWS service) comme suit :
  - L'administrateur du compte A crée un rôle IAM et attache une stratégie d'autorisation à ce rôle qui accorde des autorisations sur les ressources dans le compte A.
  - L'administrateur du compte A attache une stratégie d'approbation au rôle qui identifie le compte B comme mandataire pouvant assumer ce rôle.

- L'administrateur du compte B délègue l'autorisation d'assumer le rôle à n'importe quel utilisateur du compte B. Cela permet aux utilisateurs du compte B de créer des files d'attente pour le compte A et d'y accéder.

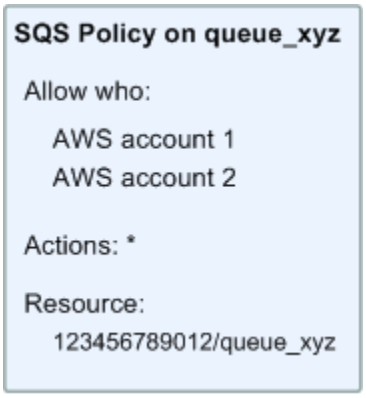
> **Note**
>
> Si vous souhaitez accorder l'autorisation d'assumer le rôle à un AWS service, le principal indiqué dans la politique de confiance peut également être un directeur AWS de service.

Pour en savoir plus sur l'utilisation d'IAM pour déléguer des autorisations, consultez Gestion des accès dans le Guide de l'utilisateur IAM.

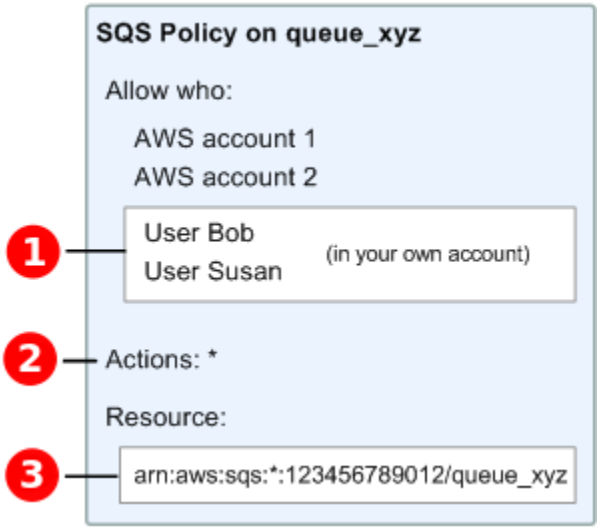
Si Amazon SQS utilise des stratégies IAM, il dispose aussi de sa propre infrastructure de stratégies. Vous pouvez utiliser une politique Amazon SQS avec une file d'attente pour spécifier quels AWS comptes ont accès à la file d'attente. Vous pouvez définir le type d'accès et les conditions (par exemple, une condition accordant des autorisations pour utiliser `SendMessage`, `ReceiveMessage` si la demande a lieu avant le 31 décembre 2010). Les actions spécifiques pour lesquelles vous pouvez accorder des autorisations représentent un sous-ensemble de la liste complète des actions Amazon SQS. Lorsque vous écrivez une stratégie Amazon SQS et que vous spécifiez * pour « autoriser toutes les actions Amazon SQS », cela signifie qu'un utilisateur peut effectuer toutes les actions de ce sous-ensemble.

Le schéma suivant illustre le concept de l'une des stratégies Amazon SQS de base, qui couvre le sous-ensemble d'actions. La politique est pour`queue_xyz`, et elle donne aux AWS comptes 1 et AWS 2 l'autorisation d'utiliser n'importe laquelle des actions autorisées avec la file d'attente spécifiée.

> **Note**
>
> La ressource spécifiée dans la politique est la suivante `123456789012/queue_xyz` : où `123456789012` est l'ID de AWS compte du compte propriétaire de la file d'attente.



Avec l'introduction d'IAM et des concepts d'utilisateurs et d'Amazon Resource Names (ARN), quelques éléments ont changé avec les stratégies SQS. Le graphique et le tableau suivants décrivent ces modifications.



1 Pour plus d'informations sur l'octroi d'autorisations aux utilisateurs de différents comptes, voir Tutoriel : déléguer l'accès entre AWS comptes à l'aide de rôles IAM dans le guide de l'utilisateur IAM.

2 Le sous-ensemble d'actions inclus dans * est plus vaste. Pour obtenir la liste des actions autorisées, consultez Autorisations d'API Amazon SQS : référence des actions et ressources.

3 Vous pouvez spécifier les ressources en indiquant leur Amazon Resource Name (ARN). Il s'agit de la méthode standard pour spécifier des ressources dans les stratégies IAM. Pour plus d'informations

sur le format ARN pour les files d'attente Amazon SQS, consultez Ressources et opérations Amazon Simple Queue Service.

Par exemple, conformément à la politique Amazon SQS décrite dans le schéma précédent, toute personne possédant les informations de sécurité du AWS compte 1 ou du AWS compte 2 peut y accéder. `queue_xyz` De plus, les utilisateurs Bob et Susan, qui appartiennent à votre compte AWS (avec l'ID `123456789012`) peuvent également accéder à la file d'attente.

Avant l'introduction d'IAM, Amazon SQS accordait automatiquement au créateur d'une file d'attente un contrôle complet sur celle-ci (c'est-à-dire l'accès à toutes les actions Amazon SQS possibles sur cette file d'attente). Ce n'est plus le cas, sauf si le créateur utilise des informations d'identification de sécurité AWS . Tout utilisateur qui dispose d'autorisations pour créer une file d'attente doit également avoir les autorisations nécessaires pour utiliser les autres actions Amazon SQS afin de pouvoir exploiter les files d'attente créées.

Dans l'exemple suivant, la stratégie autorise un utilisateur à utiliser toutes les actions Amazon SQS, mais seulement avec les files d'attente dont le nom comporte la chaîne littérale `bob_queue_` en préfixe.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:*:123456789012:bob_queue_*"
   }]
}
```

Pour plus d'informations, consultez Utilisation de politiques avec Amazon SQS et Identités (utilisateurs, groupes et rôles) dans le Guide de l'utilisateur IAM.

Spécification des éléments d'une politique : actions, effets, ressources et principaux

Pour chaque ressource Amazon Simple Queue Service, le service définit un ensemble d'actions. Pour accorder des autorisations pour ces actions, Amazon SQS définit un ensemble d'actions que vous pouvez spécifier dans une stratégie.

> **Note**
>
> Une action peut exiger des autorisations pour plusieurs actions. Lors de l'octroi des autorisations pour des actions spécifiques, vous identifiez également la ressource pour laquelle les actions sont autorisées ou refusées.

Voici les éléments les plus élémentaires d'une politique :

- Ressource : dans une politique, vous utilisez un Amazon Resource Name (ARN) pour identifier la ressource à laquelle la politique s'applique.
- Action : vous utilisez des mots clés d'action pour identifier les actions de ressource que vous voulez accorder ou refuser. Par exemple, l'autorisation `sqs:CreateQueue` permet à l'utilisateur d'effectuer l'action Amazon Simple Queue Service `CreateQueue`.
- Effet : vous spécifiez l'effet produit lorsque l'utilisateur demande l'action spécifique, qui peut être une autorisation ou un refus. Si vous n'accordez pas explicitement l'accès à une ressource, il est implicitement refusé. Vous pouvez aussi explicitement refuser l'accès à une ressource, ce que vous pouvez faire afin de vous assurer qu'un utilisateur n'y a pas accès, même si une stratégie différente accorde l'accès.
- Principal : dans les politiques basées sur une identité (politiques IAM), l'utilisateur auquel la politique est attachée est le principal implicite. Pour les politiques basées sur une ressource, vous spécifiez l'utilisateur, le compte, le service ou une autre entité qui doit recevoir les autorisations (s'applique uniquement aux politiques basées sur une ressource).

Pour en savoir plus sur la syntaxe des stratégies Amazon SQS et pour obtenir des descriptions, consultez Référence de stratégie IAM AWS dans le Guide de l'utilisateur IAM.

Pour visualiser un tableau répertoriant toutes les actions Amazon Simple Queue Service et les ressources auxquelles elles s'appliquent, consultez Autorisations d'API Amazon SQS : référence des actions et ressources.

Fonctionnement d'Amazon Simple Queue Service avec IAM

Avant d'utiliser IAM pour gérer l'accès à Amazon SQS, découvrez les fonctionnalités IAM qui peuvent être utilisées avec Amazon SQS.

Fonctionnalités IAM que vous pouvez utiliser avec Amazon Simple Queue Service

Fonction IAM	Assistance Amazon SQS
Politiques basées sur l'identité	Oui
Politiques basées sur les ressources	Oui
Actions de politique	Oui
Ressources de politique	Oui
Clés de condition de politique (spécifiques au service)	Oui
ACL	Non
ABAC (identifications dans les politiques)	Partielle
Informations d’identification temporaires	Oui
Transmission des sessions d’accès (FAS)	Oui
Fonctions de service	Oui
Rôles liés à un service	Non

Pour obtenir une vue d'ensemble de la façon dont Amazon SQS et les autres AWS services fonctionnent avec la plupart des fonctionnalités IAM, consultez les AWS services compatibles avec IAM dans le guide de l'utilisateur IAM.

Contrôle d’accès

Les listes de contrôle d’accès (ACL) vérifie quels principals (membres de compte, utilisateurs ou rôles) ont l’autorisation d’accéder à une ressource. Les listes de contrôle d’accès sont similaires aux politiques basées sur les ressources, bien qu’elles n’utilisent 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, consultez Vue d’ensemble des listes de contrôle d’accès (ACL) dans le Guide du développeur Amazon Simple Storage Service.

> **Note**
>
> Il est important de comprendre que tous Comptes AWS peuvent déléguer leurs autorisations aux utilisateurs de leurs comptes. L'accès entre comptes vous permet de partager l'accès à vos AWS ressources sans avoir à gérer d'autres utilisateurs. Pour plus d'informations sur l'accès entre comptes, consultez Activation de l'accès entre comptes dans le Guide de l'utilisateur IAM.
> Consultez Limites des politiques personnalisées d'Amazon SQS pour en savoir plus sur les autorisations et les clés de condition relatives au contenu croisé dans les stratégies personnalisées d'Amazon SQS.

Stratégies basées sur l'identité pour Amazon SQS

Prend en charge les politiques basées sur l'identité	Oui

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Création de politiques IAM dans le Guide de l'utilisateur IAM.

Avec les politiques IAM basées sur l'identité, vous pouvez spécifier des actions et ressources autorisées ou refusées, ainsi que les conditions dans lesquelles les actions sont autorisées ou refusées. Vous ne pouvez pas spécifier le principal dans une politique basée sur une identité car celle-ci s'applique à l'utilisateur ou au rôle auquel elle est attachée. Pour découvrir tous les éléments que vous utilisez dans une politique JSON, consultez Références des éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.

Exemples de stratégies basées sur l'identité pour Amazon SQS

Pour voir des exemples de stratégies Amazon SQS basées sur l'identité, consultez Bonnes pratiques en matière de politiques.

Stratégies basées sur les ressources au sein d'Amazon SQS

Prend en charge les politiques basées sur les ressources	Oui

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Des politiques basées sur les ressources sont, par exemple, les politiques de confiance de rôle IAM et des politiques de compartiment. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS

Pour permettre un accès intercompte, 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 différents Comptes AWS, un administrateur IAM du compte sécurisé doit également accorder à l'entité principale (utilisateur ou rôle) l'autorisation d'accéder à la ressource. Pour ce faire, il attache une politique basée sur une identité à l'entité. 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 plus d'informations, consultez la section Accès aux ressources entre comptes dans IAM dans le guide de l'utilisateur d'IAM.

Actions de stratégie pour Amazon SQS

Prend en charge les actions de politique	Oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Action` d'une politique JSON décrit les actions que vous pouvez utiliser pour autoriser ou refuser l'accès à une politique. Les actions de stratégie portent généralement le même nom

que l'opération AWS d'API associée. Il existe quelques exceptions, telles que les actions avec autorisations uniquement qui n'ont pas d'opération API correspondante. Certaines opérations nécessitent également plusieurs actions dans une politique. Ces actions supplémentaires sont nommées actions dépendantes.

Intégration d'actions dans une stratégie afin d'accorder l'autorisation d'exécuter les opérations associées.

Pour afficher la liste des actions Amazon SQS, consultez les Ressources définies par Amazon Simple Queue Service dans la Référence de l'autorisation de service.

Les actions de stratégie dans Amazon SQS utilisent le préfixe suivant avant l'action :

```
sqs
```

Pour indiquer plusieurs actions dans une seule déclaration, séparez-les par des virgules.

```
"Action": [
      "sqs:action1",
      "sqs:action2"
         ]
```

Pour voir des exemples de stratégies Amazon SQS basées sur l'identité, consultez Bonnes pratiques en matière de politiques.

Ressources de stratégie pour Amazon SQS

Prend en charge les ressources de politique	Oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément de politique JSON `Resource` indique le ou les objets auxquels l'action s'applique. Les instructions doivent inclure un élément `Resource` ou `NotResource`. Il est recommandé de définir

une ressource à l'aide de son Amazon Resource Name (ARN). Vous pouvez le faire pour des actions qui prennent en charge un type de ressource spécifique, connu sous la dénomination autorisations de niveau ressource.

Pour les actions qui ne sont pas compatibles avec les autorisations de niveau ressource, telles que les opérations de liste, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

```
"Resource": "*"
```

Pour afficher la liste des types de ressource Amazon SQS et leurs ARN, consultez Actions définies par Amazon Simple Queue Service dans la Référence de l'autorisation de service. Pour connaître les actions avec lesquelles vous pouvez spécifier l'ARN de chaque ressource, consultez les Ressources définies par Amazon Simple Queue Service.

Pour voir des exemples de stratégies Amazon SQS basées sur l'identité, consultez Bonnes pratiques en matière de politiques.

Clés de condition de stratégie pour Amazon SQS

Prend en charge les clés de condition de politique spécifiques au service	Oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Condition` (ou le bloc `Condition`) vous permet de spécifier des conditions lorsqu'une instruction est appliquée. L'élément `Condition` est facultatif. Vous pouvez créer des expressions conditionnelles qui utilisent des opérateurs de condition, tels que les signes égal ou inférieur à, pour faire correspondre la condition de la politique aux valeurs de la demande.

Si vous spécifiez plusieurs éléments `Condition` dans une instruction, ou plusieurs clés dans un seul élément `Condition`, AWS les évalue à l'aide d'une opération `AND` logique. Si vous spécifiez plusieurs valeurs pour une seule clé de condition, AWS évalue la condition à l'aide d'une `OR`

opération logique. Toutes les conditions doivent être remplies avant que les autorisations associées à l'instruction ne soient accordées.

Vous pouvez aussi utiliser des variables d'espace réservé quand vous spécifiez des conditions. Par exemple, vous pouvez accorder à un utilisateur IAM l'autorisation d'accéder à une ressource uniquement si elle est balisée avec son nom d'utilisateur IAM. Pour plus d'informations, consultez Éléments d'une politique IAM : variables et identifications dans le Guide de l'utilisateur IAM.

AWS prend en charge les clés de condition globales et les clés de condition spécifiques au service. Pour voir toutes les clés de condition AWS globales, voir les clés de contexte de condition AWS globales dans le guide de l'utilisateur IAM.

Pour consulter la liste des clés de condition Amazon SQS, consultez Clés de condition pour Amazon Simple Queue Service dans la Référence de l'autorisation de service. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, consultez Ressources définies par Amazon Simple Queue Service.

Pour voir des exemples de stratégies Amazon SQS basées sur l'identité, consultez Bonnes pratiques en matière de politiques.

Listes ACL dans Amazon SQS

Prend en charge les listes ACL	Non

Les listes de contrôle d'accès (ACL) vérifient quels principaux (membres de compte, utilisateurs ou rôles) ont l'autorisation d'accéder à une ressource. Les listes de contrôle d'accès sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

ABAC avec Amazon SQS

Prise en charge d'ABAC (identifications dans les politiques)	Partielle

Le contrôle d'accès basé sur les attributs (ABAC) est une politique d'autorisation qui définit des autorisations en fonction des attributs. Dans AWS, ces attributs sont appelés balises. Vous pouvez

associer des balises aux entités IAM (utilisateurs ou rôles) et à de nombreuses AWS ressources. L'étiquetage des entités et des ressources est la première étape d'ABAC. Vous concevez ensuite des politiques ABAC pour autoriser des opérations quand l'identification du principal correspond à celle de la ressource à laquelle il tente d'accéder.

L'ABAC est utile dans les environnements qui connaissent une croissance rapide et pour les cas où la gestion des politiques devient fastidieuse.

Pour contrôler l'accès basé sur des étiquettes, vous devez fournir les informations d'étiquette 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 plus d'informations sur l'ABAC, consultez Qu'est-ce que le contrôle d'accès basé sur les attributs (ABAC) ? dans le Guide de l'utilisateur IAM. Pour accéder à un didacticiel décrivant les étapes de configuration de l'ABAC, consultez Utilisation du contrôle d'accès basé sur les attributs (ABAC) dans le Guide de l'utilisateur IAM.

Utilisation d'informations d'identification temporaires avec Amazon SQS

Prend en charge les informations d'identification temporaires	Oui

Certains Services AWS ne fonctionnent pas lorsque vous vous connectez à l'aide d'informations d'identification temporaires. Pour plus d'informations, y compris celles qui Services AWS fonctionnent avec des informations d'identification temporaires, consultez Services AWS la section relative à l'utilisation d'IAM dans le guide de l'utilisateur d'IAM.

Vous utilisez des informations d'identification temporaires si vous vous connectez à l' AWS Management Console aide d'une méthode autre qu'un nom d'utilisateur et un mot de passe. Par exemple, lorsque vous accédez à AWS l'aide du lien d'authentification unique (SSO) de votre entreprise, ce processus crée automatiquement des informations d'identification temporaires. Vous créez également automatiquement des informations d'identification temporaires lorsque vous vous connectez à la console en tant qu'utilisateur, puis changez de rôle. Pour plus d'informations sur le changement de rôle, consultez Changement de rôle (console) dans le Guide de l'utilisateur IAM.

Vous pouvez créer manuellement des informations d'identification temporaires à l'aide de l' AWS API AWS CLI or. Vous pouvez ensuite utiliser ces informations d'identification temporaires pour y accéder AWS. AWS recommande de générer dynamiquement des informations d'identification temporaires au lieu d'utiliser des clés d'accès à long terme. Pour plus d'informations, consultez Informations d'identification de sécurité temporaires dans IAM.

Transférer les sessions d'accès pour Amazon SQS

Prend en charge les sessions d'accès direct (FAS)	Oui

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 une politique lors de la formulation de demandes FAS, consultez Transmission des sessions d'accès.

Fonctions du service pour Amazon SQS

Prend en charge les fonctions du service	Oui

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.

> **Warning**
>
> La modification des autorisations d'une fonction du service peut altérer la fonctionnalité d'Amazon SQS. Ne modifiez des fonctions du service que quand Amazon SQS vous le conseille.

Rôles liés à un service pour Amazon SQS

Prend en charge les rôles liés à un service	Non

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.

Pour plus d'informations sur la création ou la gestion des rôles liés à un service, consultez Services AWS qui fonctionnent avec IAM. Recherchez un service dans le tableau qui inclut un `Yes` dans la colonne Rôle lié à un service. Choisissez le lien Oui pour consulter la documentation du rôle lié à ce service.

Mises à jour des politiques gérées par Amazon SQS AWS

Pour ajouter des autorisations à des utilisateurs, des groupes et des rôles, il est plus facile d'utiliser des politiques gérées par AWS que d'écrire des politiques vous-même. 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 démarrer rapidement, vous pouvez utiliser nos politiques gérées AWS . Ces politiques couvrent des cas d'utilisation courants et sont disponibles dans votre compte AWS . Pour plus d'informations sur les politiques AWS gérées, voir les politiques AWS gérées dans le guide de l'utilisateur IAM.

AWS les services maintiennent et mettent à jour les politiques AWS gérées. Vous ne pouvez pas modifier les autorisations dans les politiques AWS gérées. Les services ajoutent parfois des autorisations supplémentaires à une politique AWS gérée pour prendre en charge de nouvelles fonctionnalités. Ce type de mise à jour affecte toutes les identités (utilisateurs, groupes et rôles) auxquelles la politique est attachée. Les services sont plus susceptibles de mettre à jour une politique AWS gérée lorsqu'une nouvelle fonctionnalité est lancée ou lorsque de nouvelles opérations sont disponibles. Les services ne suppriment pas les autorisations d'une politique AWS gérée. Les mises à jour des politiques n'endommageront donc pas vos autorisations existantes.

En outre, AWS prend en charge les politiques gérées pour les fonctions professionnelles qui couvrent plusieurs services. Par exemple, la politique ReadOnlyd'accès AWS géré fournit un accès en lecture seule à tous les AWS services et ressources. Lorsqu'un service lance une nouvelle fonctionnalité,

il AWS ajoute des autorisations en lecture seule pour les nouvelles opérations et ressources. Pour obtenir la liste des politiques de fonctions professionnelles et leurs descriptions, consultez la page politiques gérées par AWS pour les fonctions de tâche dans le Guide de l'utilisateur IAM.

AWS politique gérée : AmazonSQS FullAccess

Vous pouvez attacher la stratégie `AmazonSQSFullAccess` à vos identités Amazon SQS. Cette stratégie accorde des autorisations qui permettent un accès complet à Amazon SQS.

Pour consulter les autorisations associées à cette politique, consultez AmazonSQS FullAccess dans le manuel AWS Managed Policy Reference.

AWS politique gérée : AmazonSQS Access ReadOnly

Vous pouvez attacher la stratégie `AmazonSQSReadOnlyAccess` à vos identités Amazon SQS. Cette stratégie accorde des autorisations qui permettent un accès en lecture seule à Amazon SQS.

Pour consulter les autorisations associées à cette politique, consultez AmazonSQS ReadOnly Access dans le manuel AWS Managed Policy Reference.

Mises à jour des politiques gérées par Amazon SQS AWS

Consultez les informations relatives aux mises à jour des politiques AWS gérées pour Amazon SQS depuis que ce service a commencé à suivre ces modifications. Pour recevoir des alertes automatiques sur les modifications apportées à cette page, abonnez-vous au flux RSS de la page Historique des documents d'Amazon SQS.

Modification	Description	Date
Accès Amazon SQS ReadOnly	Amazon SQS a ajouté une nouvelle action qui vous permet de répertorier les tâches de transfert de messages les plus récentes (jusqu'à 10) dans une file d'attente source spécifique. Cette action est associée à l'opération d'API `ListMessageMoveTasks`.	9 juin 2023

Résolution des problèmes d'accès et d'identité Amazon Simple Queue Service

Utilisez les informations suivantes pour identifier et résoudre les problèmes courants que vous pouvez rencontrer lorsque vous utilisez Amazon SQS et IAM.

Rubriques

Je ne suis pas autorisé à effectuer une action dans Amazon SQS

Si vous recevez une erreur selon laquelle vous n'êtes pas autorisé à effectuer une action, vos stratégies doivent être mises à jour afin de vous permettre d'effectuer l'action.

L'exemple d'erreur suivant se produit quand l'utilisateur `mateojackson` tente d'utiliser la console pour afficher des informations détaillées sur une ressource *`my-example-widget`* fictive, mais ne dispose pas des autorisations `sqs:`*`GetWidget`* fictives.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 sqs:GetWidget on resource: my-example-widget
```

Dans ce cas, la stratégie de Mateo doit être mise à jour pour l'autoriser à accéder à la ressource *`my-example-widget`* à l'aide de l'action `sqs:`*`GetWidget`*.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d'identification de connexion.

Je ne suis pas autorisé à effectuer iam : PassRole

Si vous recevez une erreur selon laquelle vous n'êtes pas autorisé à exécuter l'action `iam:PassRole`, vos stratégies doivent être mises à jour pour vous permettre de transmettre un rôle à Amazon SQS.

Certains vous Services AWS permettent de transmettre un rôle existant à ce service au lieu de créer un nouveau rôle de service ou un rôle lié à un service. Pour ce faire, un utilisateur doit disposer des autorisations nécessaires pour transmettre le rôle au service.

L'exemple d'erreur suivant se produit lorsqu'un utilisateur IAM nommé `marymajor` essaie d'utiliser la console pour effectuer une action dans Amazon SQS. Toutefois, l'action nécessite que le service ait des autorisations accordées par un rôle de service. Mary ne dispose pas des autorisations nécessaires pour transférer le rôle au service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dans ce cas, les politiques de Mary doivent être mises à jour pour lui permettre d'exécuter l'action `iam:PassRole`.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d'identification de connexion.

Je souhaite autoriser des personnes extérieures à moi Compte AWS à accéder à mes ressources Amazon SQS

Vous pouvez créer un rôle que les utilisateurs provenant d'autres comptes ou les personnes extérieures à votre organisation pourront utiliser pour accéder à vos ressources. Vous pouvez spécifier qui est autorisé à assumer le rôle. Pour les services qui prennent en charge les politiques basées sur les ressources ou les listes de contrôle d'accès (ACL), vous pouvez utiliser ces politiques pour donner l'accès à vos ressources.

Pour en savoir plus, consultez les éléments suivants :

- Pour savoir si Amazon SQS est compatible avec ces fonctionnalités, consultez Fonctionnement d'Amazon Simple Queue Service avec IAM.
- Pour savoir comment fournir l'accès à vos ressources sur celles Comptes AWS que vous possédez, consultez la section Fournir l'accès à un utilisateur IAM dans un autre utilisateur Compte AWS que vous possédez dans le Guide de l'utilisateur IAM.
- Pour savoir comment fournir l'accès à vos ressources à des tiers Comptes AWS, consultez la section Fournir un accès à des ressources Comptes AWS détenues par des tiers dans le guide de l'utilisateur IAM.
- Pour savoir comment fournir un accès par le biais de la fédération d'identité, consultez Fournir un accès à des utilisateurs authentifiés en externe (fédération d'identité) dans le Guide de l'utilisateur IAM.

- Pour connaître la différence entre l'utilisation de rôles et de politiques basées sur les ressources pour l'accès entre comptes, consultez la section Accès aux ressources entre comptes dans IAM dans le guide de l'utilisateur d'IAM.

Utilisation de politiques avec Amazon SQS

Cette rubrique fournit des exemples de stratégies basées sur l'identité dans lesquelles un administrateur de compte peut associer des stratégies d'autorisation à des identités IAM (utilisateurs, groupes et rôles).

> **Important**
>
> Nous vous recommandons tout d'abord d'examiner les rubriques de présentation qui détaillent les concepts de base et les options disponibles pour gérer l'accès à vos ressources Amazon Simple Queue Service. Pour plus d'informations, consultez Présentation de la gestion de l'accès dans Amazon SQS.
> À l'exception de `ListQueues`, toutes les actions Amazon SQS prennent en charge les autorisations de niveau ressource. Pour plus d'informations, consultez Autorisations d'API Amazon SQS : référence des actions et ressources.

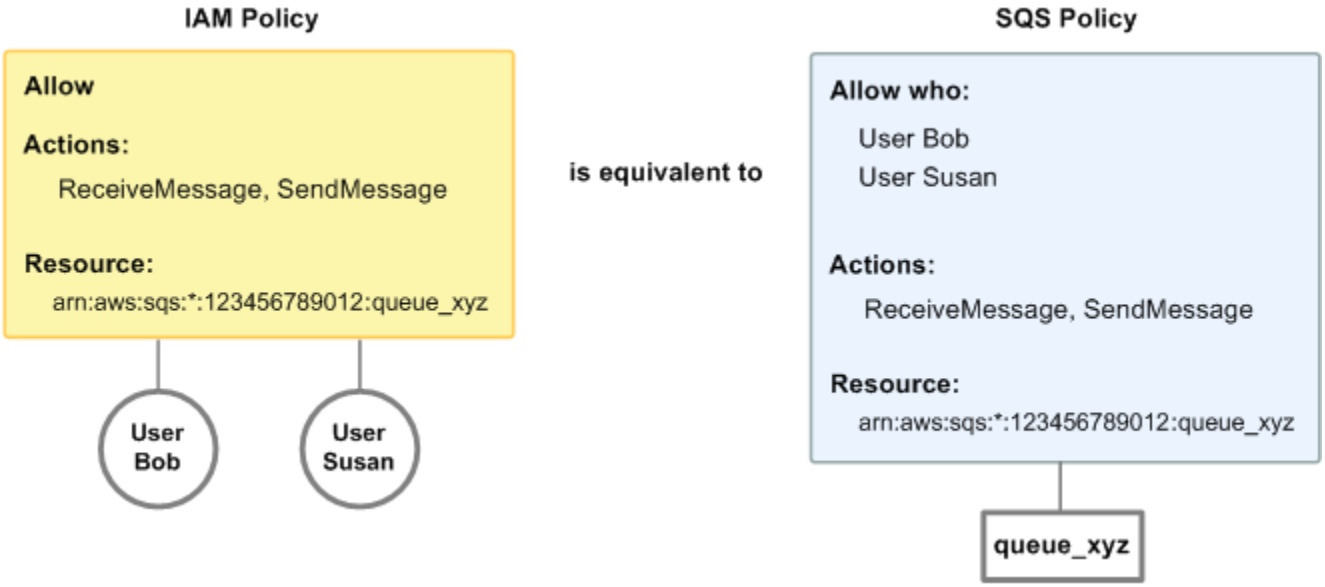
Rubriques

Utilisation des stratégies Amazon SQS et IAM

Deux options s'offrent à vous pour autoriser les utilisateurs à accéder à vos ressources Amazon SQS : le système de stratégies Amazon SQS ou le système de stratégies IAM. Vous pouvez utiliser l'un ou l'autre, ou les deux. Dans l'ensemble, vous obtenez les mêmes résultats avec l'un ou l'autre système.

Par exemple, le schéma suivant illustre une stratégie IAM et une stratégie Amazon SQS équivalente. La politique IAM accorde les droits sur Amazon `ReceiveMessage` SQS `SendMessage` et les actions relatives à la file d'attente `queue_xyz` appelée dans AWS votre compte, et la politique est attachée aux utilisateurs nommés Bob et Susan (Bob et Susan ont les autorisations indiquées dans la politique). Cette stratégie Amazon SQS donne également à Bob et à Susan des droits sur les actions `ReceiveMessage` et `SendMessage` pour cette même file d'attente.

> **Note**
>
> L'exemple suivant montre des politiques simples sans conditions. Vous pouvez spécifier une condition particulière dans l'une ou l'autre des stratégies et aboutir au même résultat.

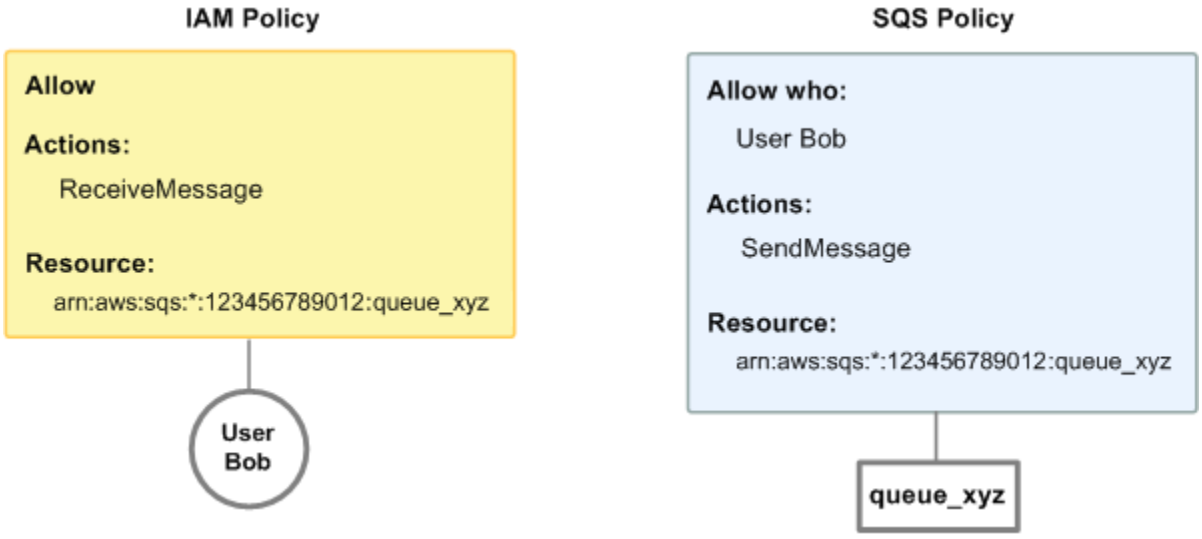


Il existe une différence majeure entre les politiques IAM et Amazon SQS : le système de politiques Amazon SQS vous permet d'accorder des autorisations à AWS d'autres comptes, contrairement à IAM.

Il vous incombe de décider si vous voulez utiliser les deux systèmes conjointement pour gérer vos autorisations. Les exemples suivants illustrent la façon dont les deux systèmes de politique interagissent.

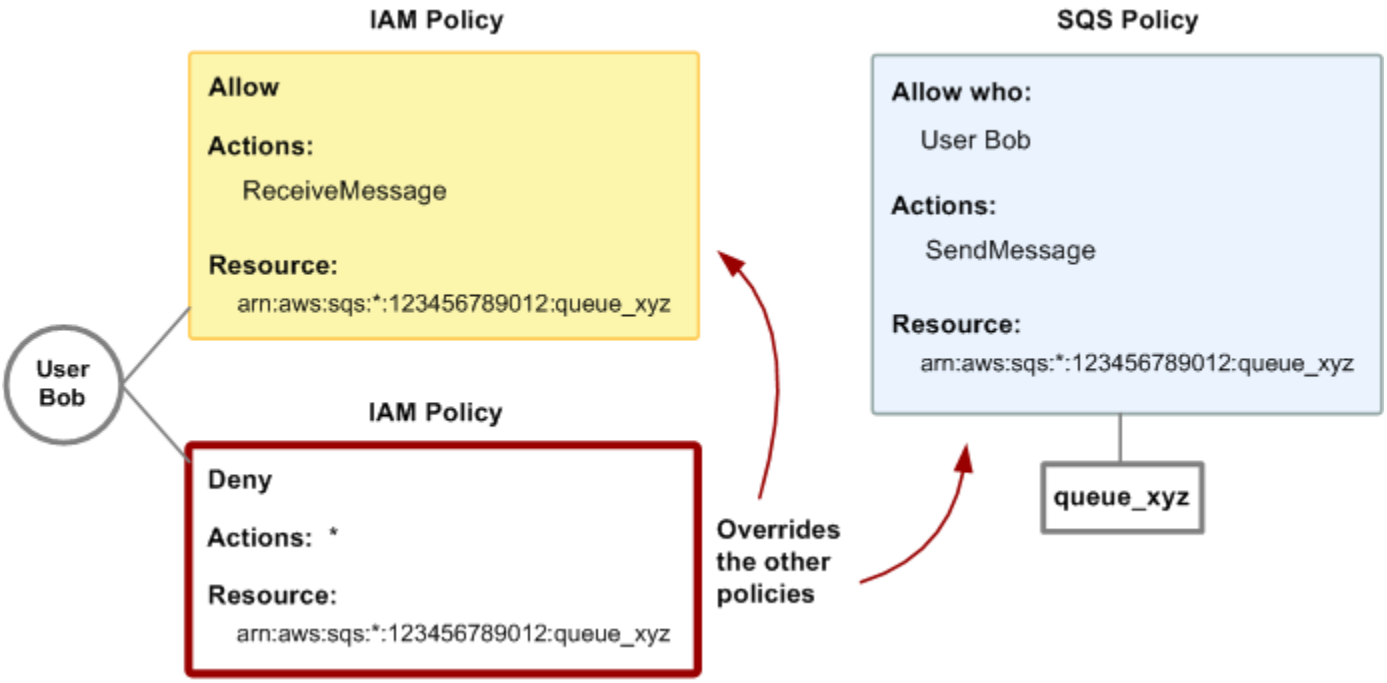
- Dans le premier exemple, Bob possède une stratégie IAM et une stratégie Amazon SQS qui s'appliquent à son compte. La stratégie IAM accorde à son compte l'autorisation d'effectuer l'action `ReceiveMessage` sur `queue_xyz`, tandis que la stratégie Amazon SQS autorise son compte

à effectuer l'action `SendMessage` sur cette même file d'attente. Le diagramme suivant illustre le concept.



Si Bob envoie une demande `ReceiveMessage` à `queue_xyz`, la stratégie IAM autorise l'action. Si Bob envoie une demande `SendMessage` à `queue_xyz`, la stratégie Amazon SQS autorise l'action.

- Dans le deuxième exemple, Bob abuse de son accès à la file d'attente `queue_xyz` de telle sorte qu'il devient nécessaire de supprimer cet accès. La méthode la plus simple consiste à ajouter une stratégie qui lui refuse l'accès à toutes les actions pour cette file d'attente. Cette stratégie prévaut sur les deux autres car une action `deny` explicite prévaut toujours sur une action `allow`. Pour plus d'informations sur la logique d'évaluation de stratégie, consultez la section Utilisation de stratégies personnalisées avec le langage de la stratégie d'accès Amazon SQS. Le diagramme suivant illustre le concept.



Vous pouvez également ajouter une instruction supplémentaire à la stratégie Amazon SQS, qui refuse à Bob tout type d'accès à la file d'attente. Cette approche a le même effet que l'ajout d'une stratégie IAM refusant à Bob l'accès à la file d'attente. Pour obtenir des exemples de stratégies couvrant les actions et ressources Amazon SQS, consultez Exemples de base de stratégies Amazon SQS. Pour plus d'informations sur l'écriture de stratégies Amazon SQS, consultez Utilisation de stratégies personnalisées avec le langage de la stratégie d'accès Amazon SQS.

Autorisations requises pour utiliser la console Amazon SQS

Un utilisateur désireux d'utiliser la console Amazon SQS doit disposer d'un ensemble minimal d'autorisations pour utiliser les files d'attente Amazon SQS dans son Compte AWS. Par exemple, l'utilisateur doit avoir l'autorisation d'appeler l'action `ListQueues` pour répertorier les files d'attente ou l'action `CreateQueue` pour créer des files d'attente. Outre les autorisations Amazon SQS pour abonner une file d'attente Amazon SQS à une rubrique Amazon SNS, la console exige également des autorisations pour les actions Amazon SNS.

Si vous créez une stratégie IAM plus restrictive que les autorisations minimales requises, la console peut ne pas fonctionner comme prévu pour les utilisateurs dotés de la stratégie IAM.

Il n'est pas nécessaire d'accorder des autorisations de console minimales aux utilisateurs qui appellent uniquement les actions AWS CLI ou Amazon SQS.

Exemples de stratégies basées sur l'identité pour Amazon SQS

Par défaut, les utilisateurs et les rôles ne sont pas autorisés à créer ou à modifier des ressources Amazon SQS. Ils ne peuvent pas non plus effectuer de tâches à l'aide de l'API AWS Management Console, AWS Command Line Interface (AWS CLI) ou de AWS l'API. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM. L'administrateur peut ensuite ajouter les politiques IAM aux rôles et les utilisateurs peuvent assumer les rôles.

Pour apprendre à créer une politique basée sur l'identité IAM à l'aide de ces exemples de documents de politique JSON, consultez Création de politiques dans l'onglet JSON dans le Guide de l'utilisateur IAM.

Pour plus de détails sur les actions et les types de ressources définis par Amazon SQS, y compris le format des ARN pour chacun des types de ressources, consultez Actions, ressources et clés de condition pour Amazon Simple Queue Service dans la Référence de l'autorisation de service.

> **Note**
>
> Lorsque vous configurez des hooks de cycle de vie pour Amazon EC2 Auto Scaling, vous n'avez pas besoin d'écrire de stratégie pour envoyer des messages à une file d'attente Amazon SQS. Pour plus d'informations, consultez Amazon EC2 Auto Scaling Lifecycle Hooks dans le guide de l'utilisateur Amazon EC2.

Rubriques

Bonnes pratiques en matière de politiques

Les stratégies basées sur l'identité déterminent si une personne peut créer, consulter ou supprimer des ressources Amazon SQS dans votre compte. Ces actions peuvent entraîner des frais pour votre Compte AWS. Lorsque vous créez ou modifiez des politiques basées sur l'identité, suivez ces instructions et recommandations :

- Commencez AWS par les politiques gérées et passez aux autorisations du moindre privilège : pour commencer à accorder des autorisations à vos utilisateurs et à vos charges de travail, utilisez les politiques AWS gérées qui accordent des autorisations pour de nombreux cas d'utilisation courants. Ils sont disponibles dans votre Compte AWS. Nous vous recommandons de réduire davantage les autorisations en définissant des politiques gérées par le AWS client spécifiques à vos cas d'utilisation. Pour plus d'informations, consultez politiques gérées par AWS ou politiques gérées par AWS pour les activités professionnelles dans le Guide de l'utilisateur IAM.
- Accorder les autorisations de moindre privilège : lorsque vous définissez des autorisations avec des politiques 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. Pour plus d'informations sur l'utilisation de IAM pour appliquer des autorisations, consultez politiques et autorisations dans IAM dans le Guide de l'utilisateur IAM.
- Utiliser des conditions dans les politiques IAM pour restreindre davantage l'accès : vous pouvez ajouter une condition à vos politiques afin de limiter l'accès aux actions et aux ressources. Par exemple, vous pouvez écrire une condition de politique 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 si elles sont utilisées par le biais d'un service spécifique Service AWS, tel que AWS CloudFormation. Pour plus d'informations, consultez Conditions pour éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.
- Utilisez IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles : IAM Access Analyzer valide les politiques nouvelles et existantes de manière à ce que les politiques IAM respectent le langage de politique IAM (JSON) et les bonnes pratiques IAM. IAM Access Analyzer fournit plus de 100 vérifications de politiques et des recommandations exploitables pour vous aider à créer des politiques sécurisées et fonctionnelles. Pour plus d'informations, consultez Validation de politique IAM Access Analyzer dans le Guide de l'utilisateur IAM.
- Exiger l'authentification multifactorielle (MFA) : si vous avez un scénario qui nécessite des utilisateurs IAM ou un utilisateur root, activez l'authentification MFA pour une sécurité accrue.

Compte AWS Pour exiger le MFA lorsque des opérations d'API sont appelées, ajoutez des conditions MFA à vos politiques. Pour plus d'informations, consultez Configuration de l'accès aux API protégé par MFA dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur les bonnes pratiques dans IAM, consultez Bonnes pratiques de sécurité dans IAM dans le Guide de l'utilisateur IAM.

Utilisation de la console Amazon SQS

Pour accéder à la console Amazon Simple Queue Service, vous devez disposer d'un ensemble minimum d'autorisations. Ces autorisations doivent vous permettre de répertorier et de consulter les informations relatives aux ressources Amazon SQS de votre. Compte AWS Si vous créez une stratégie basée sur l'identité qui est plus restrictive que l'ensemble minimum d'autorisations requis, la console ne fonctionnera pas comme prévu pour les entités (utilisateurs ou rôles) tributaires de cette stratégie.

Il n'est pas nécessaire d'accorder des autorisations de console minimales aux utilisateurs qui appellent uniquement l'API AWS CLI ou l' AWS API. Autorisez plutôt l'accès à uniquement aux actions qui correspondent à l'opération d'API qu'ils tentent d'effectuer.

Pour garantir que les utilisateurs et les rôles peuvent toujours utiliser la console Amazon SQS, associez également la politique gérée par Amazon `AmazonSQSReadOnlyAccess` AWS SQS aux entités. Pour plus d'informations, consultez Ajout d'autorisations à un utilisateur dans le Guide de l'utilisateur IAM.

Autorisation accordée aux utilisateurs pour afficher leurs propres autorisations

Cet exemple montre comment créer une politique qui permet aux utilisateurs IAM d'afficher les politiques en ligne et gérées attachées à leur identité d'utilisateur. Cette politique inclut les autorisations permettant d'effectuer cette action sur la console ou par programmation à l'aide de l'API AWS CLI or AWS .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
```

```
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Autoriser un utilisateur à créer des files d'attente

Dans l'exemple suivant, nous créons une stratégie qui permet à Bob d'accéder à toutes les actions Amazon SQS, mais seulement avec les files d'attente dont le nom comporte la chaîne littérale `alice_queue_` en préfixe.

Amazon SQS n'accorde pas automatiquement au créateur d'une file d'attente les autorisations de l'utiliser. Par conséquent, dans la stratégie IAM, nous devons explicitement accorder à Bob les autorisations d'utiliser toutes les actions Amazon SQS en plus de l'action `CreateQueue`.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:*:123456789012:alice_queue_*"
   }]
}
```

Permettre aux développeurs d'écrire des messages dans une file d'attente partagée

Dans l'exemple suivant, nous créons un groupe pour les développeurs et y attachons une politique qui permet au groupe d'utiliser l'`SendMessage`action Amazon SQS, mais uniquement avec la file d'attente qui appartient à la file spécifiée Compte AWS et qui est nommée. `MyCompanyQueue`

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "sqs:SendMessage",
      "Resource": "arn:aws:sqs:*:123456789012:MyCompanyQueue"
   }]
}
```

Vous pouvez utiliser * au lieu de `SendMessage` pour attribuer les actions suivantes à un mandataire sur une file d'attente partagée : `ChangeMessageVisibility`, `DeleteMessage`, `GetQueueAttributes`, `GetQueueUrl`, `ReceiveMessage` et `SendMessage`.

> **Note**
>
> Bien que * comprenne l'accès fourni par d'autres types d'autorisation, Amazon SQS examine les autorisations séparément. Par exemple, il est possible d'accorder à la fois les autorisations * et `SendMessage` à un utilisateur, même si le symbole * inclut l'accès fourni par `SendMessage`.
>
> Ce concept s'applique également quand vous supprimez une autorisation. Si un mandataire dispose uniquement d'une autorisation *, toute demande de suppression de l'autorisation `SendMessage` ne signifie pas que l'utilisateur peut tout faire, sauf cette action. Au lieu de cela, cette demande n'a aucun effet, car le mandataire n'a aucune autorisation `SendMessage` explicite. Pour attribuer uniquement l'autorisation `ReceiveMessage` au mandataire, commencez par ajouter l'autorisation `ReceiveMessage`, puis supprimez l'autorisation *.

Permettre aux gestionnaires d'obtenir la taille générale des files d'attente

Dans l'exemple suivant, nous créons un groupe pour les responsables et y attachons une politique qui permet au groupe d'utiliser l'`GetQueueAttributes`action Amazon SQS avec toutes les files d'attente appartenant au compte spécifié. AWS

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": "sqs:GetQueueAttributes",
      "Resource": "*"
   }]
}
```

Autoriser un partenaire à envoyer des messages à une file d'attente spécifique

Vous pouvez accomplir cette tâche à l'aide d'une stratégie Amazon SQS ou IAM. Si votre partenaire dispose d'une politique Amazon SQS Compte AWS, il peut être plus facile d'utiliser une politique Amazon SQS. Cependant, tout utilisateur de l'entreprise du partenaire qui possède les informations de AWS sécurité peut envoyer des messages à la file d'attente. Si vous souhaitez limiter l'accès à un utilisateur ou une application en particulier, vous devez traiter le partenaire comme un utilisateur de votre propre entreprise et utiliser une stratégie IAM au lieu d'une stratégie Amazon SQS.

L'exemple suivant effectue les actions suivantes :

1. Créez un groupe appelé WidgetCo pour représenter l'entreprise partenaire.
2. Création d'un utilisateur pour la personne ou l'application spécifique qui a besoin d'un accès dans l'entreprise du partenaire.
3. Ajoutez l'utilisateur au groupe .
4. Association d'une stratégie qui donne au groupe l'accès à l'action `SendMessage`, mais uniquement pour la file d'attente `WidgetPartnerQueue`.

```
{
   "Version": "2012-10-17",
   "Statement": [{
         "Effect": "Allow",
         "Action": "sqs:SendMessage",
         "Resource": "arn:aws:sqs:*:123456789012:WidgetPartnerQueue"
   }]
}
```

Exemples de base de stratégies Amazon SQS

Cette section présente des exemples de stratégies pour les cas d'utilisation Amazon SQS les plus courants.

Vous pouvez utiliser la console pour vérifier les effets de chaque politique lorsque vous les associez à l'utilisateur. Au départ, l'utilisateur n'a pas les autorisations requises et ne peut donc effectuer aucune action dans la console. A mesure que vous lui associez des stratégies, vous pouvez vérifier que l'utilisateur peut exécuter diverses actions dans la console.

> **Note**
>
> Nous vous recommandons d'utiliser deux fenêtres de navigateur : l'une pour accorder des autorisations et l'autre pour vous connecter à l' AWS Management Console aide des informations d'identification de l'utilisateur afin de vérifier les autorisations que vous lui accordez.

Exemple 1 : accorder une autorisation à une Compte AWS

L'exemple de politique suivant accorde à Compte AWS number `111122223333` l'`SendMessage`autorisation pour la file d'attente nommée `444455556666/queue1` dans la région USA Est (Ohio).

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_SendMessage",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "111122223333"
         ]
      },
      "Action": "sqs:SendMessage",
      "Resource": "arn:aws:sqs:us-east-2:444455556666:queue1"
   }]
}
```

Exemple 2 : accorder deux autorisations à une Compte AWS

L'exemple de politique suivant accorde à la 111122223333 fois le Compte AWS numéro SendMessage et l'ReceiveMessageautorisation pour la file d'attente nommée444455556666/queue1.

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_Send_Receive",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "111122223333"
         ]
      },
      "Action": [
         "sqs:SendMessage",
         "sqs:ReceiveMessage"
      ],
      "Resource": "arn:aws:sqs:*:444455556666:queue1"
   }]
}
```

Exemple 3 : accorder toutes les autorisations à deux Comptes AWS

L'exemple de politique suivant accorde deux Comptes AWS numéros différents (111122223333et444455556666) l'autorisation d'utiliser toutes les actions auxquelles Amazon SQS autorise un accès partagé pour la file d'attente nommée 123456789012/queue1 dans la région USA Est (Ohio).

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_AllActions",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "111122223333",
            "444455556666"
```

```
         ]
      },
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:us-east-2:123456789012:queue1"
   }]
}
```

Exemple 4 : Accorder des autorisations inter-comptes à un rôle et à un nom d'utilisateur

L'exemple de politique suivant accorde`role1`, `username1` sous un Compte AWS numéro`111122223333`, l'autorisation entre comptes d'utiliser toutes les actions auxquelles Amazon SQS autorise un accès partagé pour la file d'attente `123456789012/queue1` nommée dans la région USA Est (Ohio).

Les autorisations intercompte ne s'appliquent pas aux actions suivantes :

- AddPermission
- CancelMessageMoveTask
- CreateQueue
- DeleteQueue
- ListMessageMoveTask
- ListQueues
- ListQueueTags
- RemovePermission
- SetQueueAttributes
- StartMessageMoveTask
- TagQueue
- UntagQueue

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_AllActions",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
```

```
            "arn:aws:iam::111122223333:role/role1",
            "arn:aws:iam::111122223333:user/username1"
         ]
      },
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:us-east-2:123456789012:queue1"
   }]
}
```

Exemple 5 : Accorder une autorisation à tous les utilisateurs

L'exemple de stratégie suivant accorde à tous les utilisateurs (anonymes) l'autorisation `ReceiveMessage` pour la file d'attente dénommée `111122223333/queue1`.

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_AnonymousAccess_ReceiveMessage",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "sqs:ReceiveMessage",
      "Resource": "arn:aws:sqs:*:111122223333:queue1"
   }]
}
```

Exemple 6 : Accorder une autorisation limitée dans le temps à tous les utilisateurs

L'exemple de stratégie suivant accorde à tous les utilisateurs (anonymes) l'autorisation `ReceiveMessage` pour la file d'attente dénommée `111122223333/queue1`, mais seulement entre 12 h (midi) et 15 h le 31 janvier 2009.

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_AnonymousAccess_ReceiveMessage_TimeLimit",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "sqs:ReceiveMessage",
      "Resource": "arn:aws:sqs:*:111122223333:queue1",
      "Condition" : {
```

```
         "DateGreaterThan" : {
            "aws:CurrentTime":"2009-01-31T12:00Z"
         },
         "DateLessThan" : {
            "aws:CurrentTime":"2009-01-31T15:00Z"
         }
      }
   }]
}
```

Exemple 7 : Accorder toutes les autorisations à tous les utilisateurs d'une plage d'adresses CIDR

L'exemple de stratégie suivant accorde à tous les utilisateurs (anonymes) l'autorisation d'utiliser toutes les actions Amazon SQS qui peuvent être partagées pour la file d'attente nommée `111122223333/queue1`, mais uniquement si la demande provient de la plage d'adresses CIDR `192.0.2.0/24`.

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_AnonymousAccess_AllActions_AllowlistIP",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:*:111122223333:queue1",
      "Condition" : {
         "IpAddress" : {
            "aws:SourceIp":"192.0.2.0/24"
         }
      }
   }]
}
```

Exemple 8 : Ajouter les utilisateurs de différentes plages d'adresses CIDR à une liste d'autorisations ou à une liste de blocage pour leur permettre ou les empêcher d'effectuer une action

L'exemple de stratégie suivant comporte deux instructions :

- La première instruction accorde à tous les utilisateurs (anonymes) de la plage d'adresses CIDR `192.0.2.0/24` (à l'exception de `192.0.2.188`) l'autorisation d'utiliser l'action `SendMessage` pour la file d'attente dénommée `111122223333/queue1`.

- La deuxième instruction empêche tous les utilisateurs (anonymes) de la plage d'adresses CIDR `12.148.72.0/23` d'utiliser la file d'attente en les ajoutant à une liste de blocage.

```
{
   "Version": "2012-10-17",
   "Id": "Queue1_Policy_UUID",
   "Statement": [{
      "Sid":"Queue1_AnonymousAccess_SendMessage_IPLimit",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "sqs:SendMessage",
      "Resource": "arn:aws:sqs:*:111122223333:queue1",
      "Condition" : {
         "IpAddress" : {
            "aws:SourceIp":"192.0.2.0/24"
         },
         "NotIpAddress" : {
            "aws:SourceIp":"192.0.2.188/32"
         }
      }
   }, {
      "Sid":"Queue1_AnonymousAccess_AllActions_IPLimit_Deny",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "sqs:*",
      "Resource": "arn:aws:sqs:*:111122223333:queue1",
      "Condition" : {
         "IpAddress" : {
            "aws:SourceIp":"12.148.72.0/23"
         }
      }
   }]
}
```

Utilisation de stratégies personnalisées avec le langage de la stratégie d'accès Amazon SQS

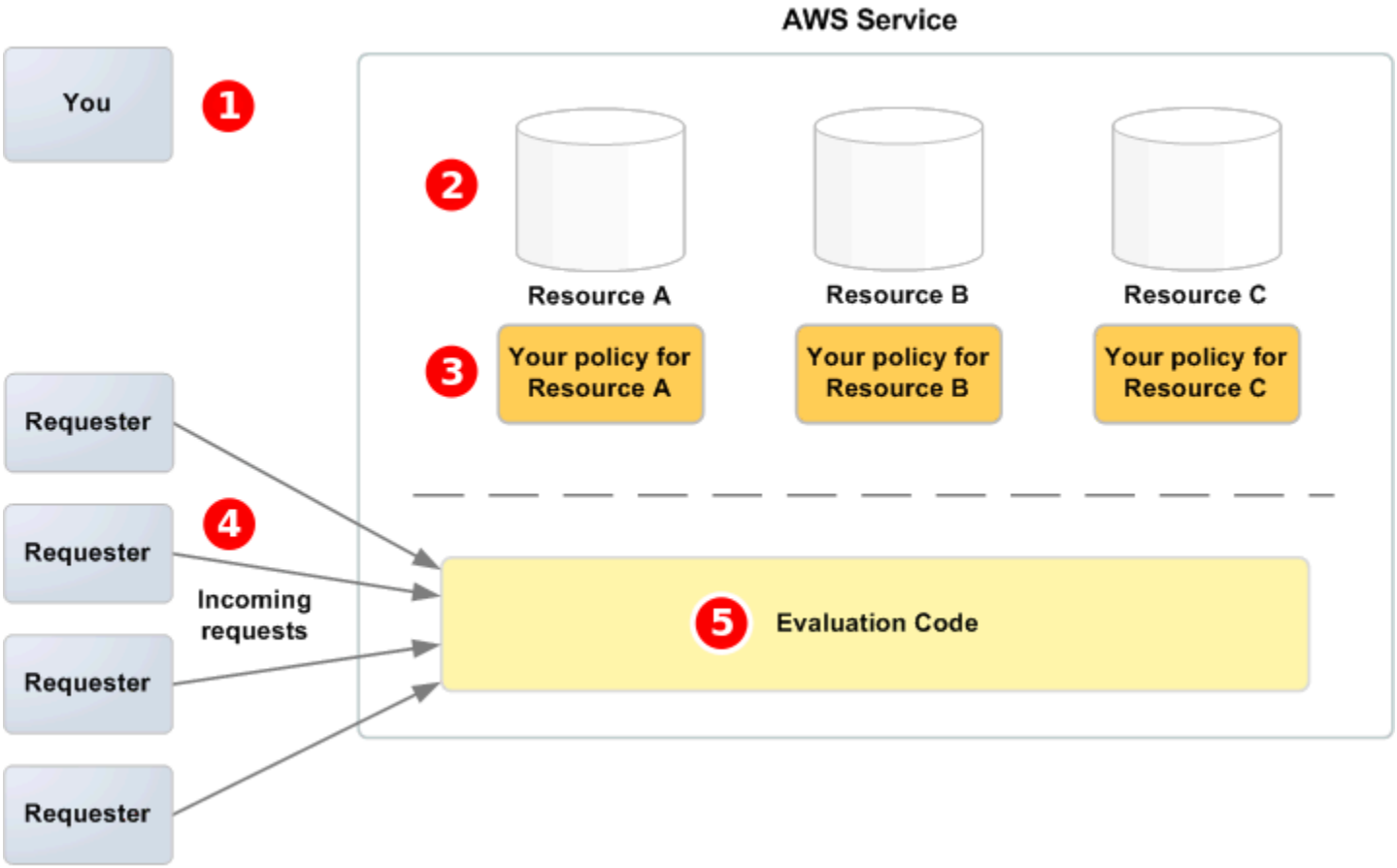
Si vous souhaitez autoriser l'accès à Amazon SQS uniquement sur la base d'un Compte AWS identifiant et d'autorisations de base (par exemple pour `SendMessage`ou `ReceiveMessage`), vous n'avez pas besoin de rédiger vos propres politiques. Vous pouvez simplement utiliser l'action `AddPermission` Amazon SQS.

Si vous souhaitez refuser ou autoriser explicitement l'accès en fonction de conditions plus spécifiques (telles que l'heure d'arrivée de la demande ou l'adresse IP du demandeur), vous devez rédiger vos propres politiques Amazon SQS et les télécharger dans AWS le système à l'aide de l'action Amazon SQS. `SetQueueAttributes`

Rubriques

Architecture de contrôle d'accès Amazon SQS

Le schéma suivant décrit le système de contrôle d'accès pour vos ressources Amazon SQS.



1

Vous-même, le propriétaire de la ressource.

2

Vos ressources contenues dans le AWS service (par exemple, les files d'attente Amazon SQS).

3

Vos stratégies. Il est recommandé d'avoir une stratégie par ressource. Le AWS service fournit une API que vous utilisez pour télécharger et gérer vos politiques.

4

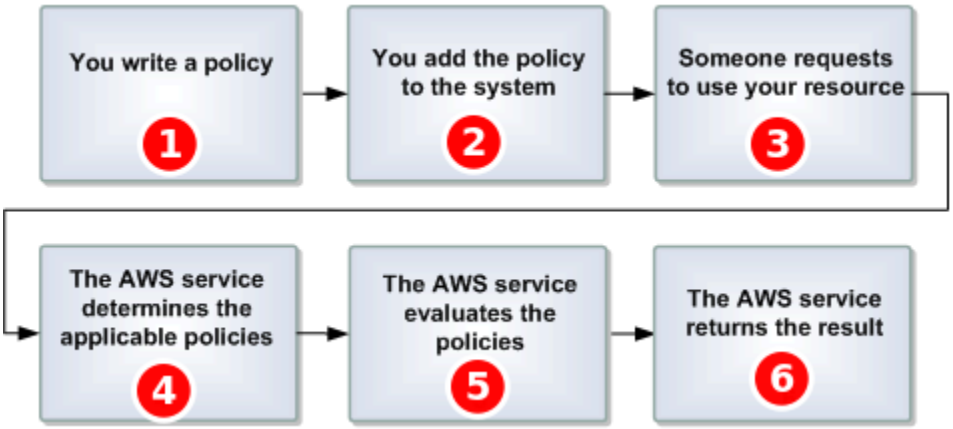
Les demandeurs et leurs demandes entrantes au service AWS .

5

Code d'évaluation du langage de la stratégie d'accès. Il s'agit de l'ensemble de code du AWS service qui évalue les demandes entrantes par rapport aux politiques applicables et détermine si le demandeur est autorisé à accéder à la ressource.

Flux de travail des processus de contrôle d'accès Amazon SQS

Le schéma suivant décrit le flux de travail général du contrôle d'accès avec le langage de la politique d'accès Amazon SQS.



1

Vous écrivez une stratégie Amazon SQS pour votre file d'attente.

2

Vous téléchargez votre politique sur AWS. Le AWS service fournit une API que vous utilisez pour télécharger vos politiques. Par exemple, vous utilisez l'action Amazon SQS `SetQueueAttributes` pour importer une stratégie pour une file d'attente Amazon SQS spécifique.

3

Quelqu'un envoie une demande d'utilisation de votre file d'attente Amazon SQS.

4

Amazon SQS examine toutes les stratégies Amazon SQS disponibles et détermine lesquelles sont applicables.

5

Amazon SQS évalue les stratégies et détermine si le demandeur est autorisé à utiliser votre file d'attente.

6

En fonction du résultat de l'évaluation de la stratégie, Amazon SQS renvoie une erreur `Access denied` au demandeur ou continue à traiter la demande.

Concepts clés du langage de la stratégie d'accès Amazon SQS

Pour écrire vos propres stratégies, vous devez être familiarisé avec le langage JSON et un certain nombre de concepts clés.

Autorisation

Résultat d'une Instruction dont l'Effet est défini sur `allow`.

Action

Activité que le Principal est autorisé à effectuer, généralement une demande à AWS.

Default-deny

Résultat d'une Instruction ne disposant d'aucun paramètre Autorisation ou Explicit-deny.

Condition

Tout détail ou restriction concernant une Autorisation. Les conditions typiques sont liées à la date et à l'heure, ainsi qu'aux adresses IP.

Effet

Résultat que vous souhaitez que la Instruction d'une Stratégie renvoie au moment de l'évaluation. Vous spécifiez la valeur `deny` ou `allow` lorsque vous écrivez la déclaration de stratégie. Trois résultats sont possibles lors de l'évaluation de stratégie : Default-deny, Autorisation ou Explicit-deny.

Explicit-deny

Résultat d'une Instruction dont l'Effet est défini sur `deny`.

Evaluation

Processus utilisé par Amazon SQS pour déterminer si une demande entrante doit être refusée ou autorisée en fonction d'une Stratégie.

Emetteur

Utilisateur qui écrit une Stratégie pour accorder des autorisations à une ressource. Par définition, l'émetteur est toujours le propriétaire de la ressource. AWS n'autorise pas les utilisateurs d'Amazon SQS à créer des politiques pour des ressources dont ils ne sont pas propriétaires.

Clé

Caractéristique spécifique à la base d'une restriction d'accès.

Autorisation

Concept consistant à autoriser ou refuser l'accès à une ressource à l'aide d'une Condition et d'une Clé.

Stratégie

Document jouant le rôle de conteneur pour une ou plusieurs déclarations.

Amazon SQS utilise la stratégie pour déterminer s'il convient d'autoriser un utilisateur à accéder à une ressource.

Principal

Utilisateur qui reçoit l'Autorisation dans la Stratégie.

Ressource

Objet auquel le Principal demande l'accès.

Instruction

Description formelle d'une autorisation unique, écrite dans le langage de la stratégie d'accès, dans le cadre d'un document de Stratégie plus vaste.

Demandeur

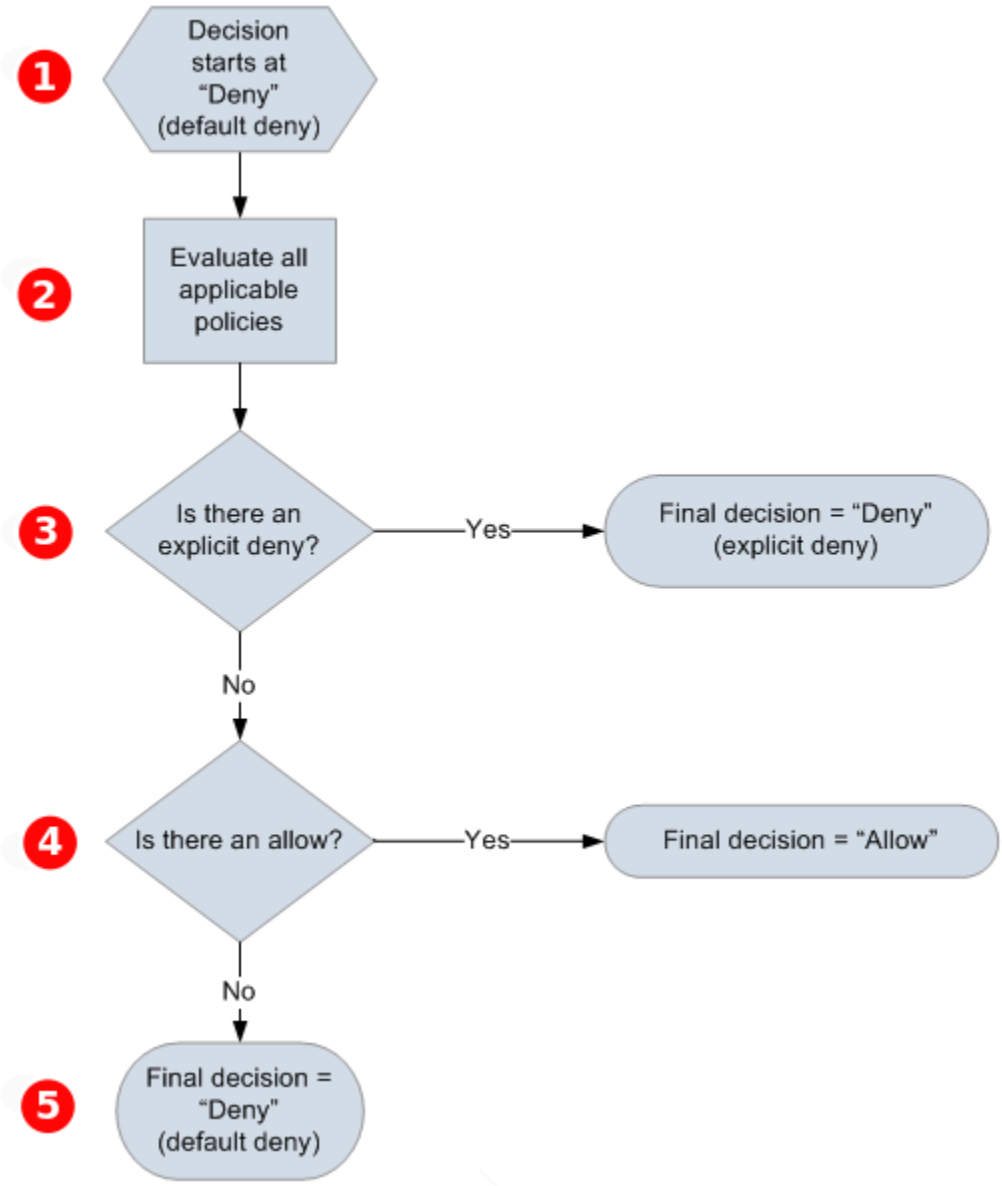
Utilisateur qui envoie une demande d'accès à une Ressource.

Logique d'évaluation du langage de la stratégie d'accès Amazon SQS

Au moment de l'évaluation, Amazon SQS détermine si une demande d'un utilisateur qui n'est pas le propriétaire de la ressource doit être autorisée ou refusée. La logique d'évaluation suit plusieurs règles de base :

- Par défaut, toutes les demandes d'utilisation de votre ressource ne provenant pas de vous-même sont refusées.
- La valeur Autorisation prévaut sur Default-deny.
- La valeur Explicit-deny prévaut sur allow.
- L'ordre dans lequel les stratégies sont évaluées n'a pas d'importance.

Le schéma suivant décrit en détail la façon dont Amazon SQS évalue les décisions relatives aux autorisations d'accès.



1. La décision commence par un refus par défaut (default-deny).

2. Le code d'application évalue toutes les stratégies qui s'appliquent à la demande (en se basant sur la ressource, le mandataire, l'action et les conditions). L'ordre dans lequel le code d'application évalue les stratégies n'a pas d'importance.

3. Le code d'application recherche une instruction explicit-deny qui peut s'appliquer à la demande. S'il en trouve une, le code d'application renvoie une décision de type deny (refus) et le processus se termine.

4. En l'absence d'instruction explicit-deny (refus explicite), le code d'application recherche des instructions allow (autorisation) pouvant s'appliquer à la demande. S'il en trouve une, il renvoie une décision de type allow (autoriser) et le processus se termine (le service continue à traiter la demande).

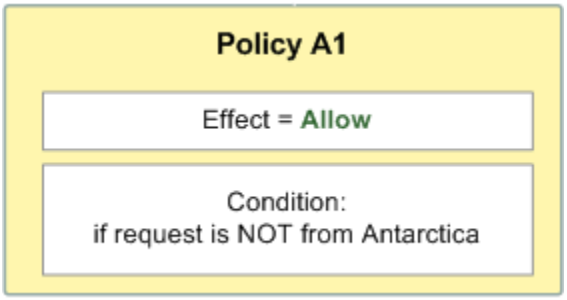
5. Si aucune instruction allow n'est détectée, la décision finale est un refus (deny). En l'absence de refus explicite (explicit-deny) ou d'autorisation (allow), nous parlons d'un refus par défaut (default-deny).

Relations entre les refus explicites et les refus par défaut dans le langage de la stratégie d'accès Amazon SQS

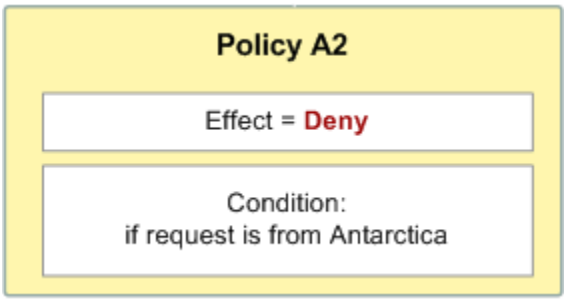
Si une stratégie Amazon SQS ne s'applique pas directement à une demande, celle-ci se conclut par un Default-deny. Par exemple, si un utilisateur demande l'autorisation d'utiliser Amazon SQS, mais que la seule stratégie qui s'applique à l'utilisateur indique qu'il peut utiliser DynamoDB, la demande se conclut par un default-deny.

Si une condition de la déclaration n'est pas respectée, la demande se conclut par un default-deny. Si toutes les conditions d'une déclaration sont respectées, la demande se conclut par une décision de type Autorisation ou Explicit-deny, en fonction de la valeur de l'élément Effet de la stratégie. Les stratégies ne spécifient pas comment procéder si une condition n'est pas respectée. Le résultat par défaut est donc un default-deny dans ce cas. Supposons par exemple que vous souhaitez refuser les demandes provenant de l'Antarctique. Vous créez une stratégie Policy A1 qui autorise une

demande uniquement si elle ne provient pas de l'Antarctique. Le schéma suivant illustre la stratégie Amazon SQS.

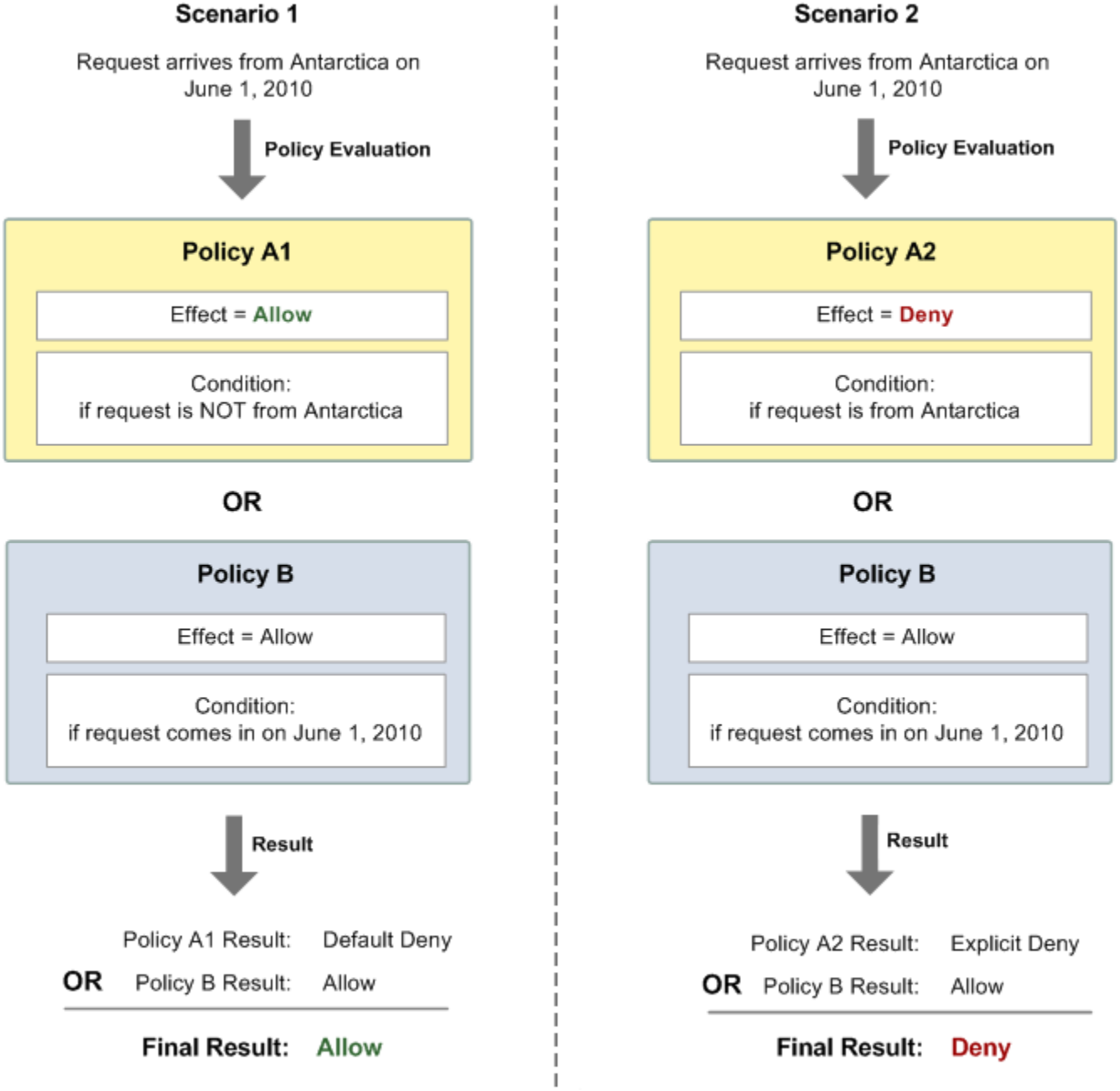


Si un utilisateur envoie une demande depuis les États-Unis, la condition est respectée (la demande ne provient pas de l'Antarctique) et la demande se conclut par une décision allow. Par contre, si un utilisateur envoie une demande depuis l'Antarctique, la condition n'est pas respectée et la demande se conclut par défaut par un default-deny. Vous pouvez modifier le résultat et le faire passer en explicit-deny en écrivant une stratégie Policy A2 qui refuse explicitement toute demande provenant de l'Antarctique. Le schéma suivant illustre la politique.



Si un utilisateur envoie une demande depuis l'Antarctique, la condition est remplie et la demande se conclut par un explicit-deny.

Il est important de bien faire la distinction entre un default-deny et un explicit-deny, car une décision allow peut prévaloir sur le premier mais pas sur le second. Par exemple, la stratégie Policy B autorise les demandes si elles arrivent le 1er juin 2010. Le schéma suivant compare l'association de cette stratégie avec une stratégie Policy A1 et une stratégie Policy A2.



Dans le scénario 1, la stratégie Policy A1 se conclut par un default-deny et la stratégie Policy B par un allow, car la stratégie autorise les demandes qui arrivent le 1er juin 2010. L'autorisation allow de Policy B remplace le refus par défaut (default-deny) de Policy A1 et, par conséquent, la demande est autorisée.

Dans le scénario 2, la stratégie Policy B2 génère un explicit-deny et la stratégie Policy B génère une décision allow. Le refus explicite (explicit-deny) de Policy A2 remplace l'autorisation (allow) de Policy B et, par conséquent, la demande est refusée.

Limites des politiques personnalisées d'Amazon SQS

Accès intercomptes

Les autorisations intercompte ne s'appliquent pas aux actions suivantes :

- `AddPermission`
- `CancelMessageMoveTask`
- `CreateQueue`
- `DeleteQueue`
- `ListMessageMoveTask`
- `ListQueues`
- `ListQueueTags`
- `RemovePermission`
- `SetQueueAttributes`
- `StartMessageMoveTask`
- `TagQueue`
- `UntagQueue`

Clés de condition

Actuellement, Amazon SQS prend en charge uniquement un sous-ensemble limité des clés de condition disponibles dans IAM. Pour plus d'informations, consultez Autorisations d'API Amazon SQS : référence des actions et ressources.

Exemples de langage de la stratégie d'accès Amazon SQS personnalisé

Voici des exemples classiques de stratégies d'accès Amazon SQS.

Exemple 1 : Accorder une autorisation à un compte

L'exemple de stratégie Amazon SQS suivant accorde au compte Compte AWS 111122223333 l'autorisation d'envoyer et de recevoir la file d'attente `queue2` détenue par le compte Compte AWS 444455556666.

```
{
   "Version": "2012-10-17",
   "Id": "UseCase1",
```

```
   "Statement" : [{
      "Sid": "1",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "111122223333"
         ]
      },
      "Action": [
         "sqs:SendMessage",
         "sqs:ReceiveMessage"
      ],
      "Resource": "arn:aws:sqs:us-east-2:444455556666:queue2"
   }]
}
```

Exemple 2 : Accorder une autorisation à un ou plusieurs comptes

L'exemple suivant de politique Amazon SQS donne un ou plusieurs Comptes AWS accès aux files d'attente détenues par votre compte pendant une période spécifique. Il est nécessaire d'écrire cette stratégie et de la télécharger dans Amazon SQS à l'aide de l'action `SetQueueAttributes`, car l'action `AddPermission` ne permet pas de spécifier une restriction de durée lors de l'octroi de l'accès à une file d'attente.

```
{
   "Version": "2012-10-17",
   "Id": "UseCase2",
   "Statement" : [{
      "Sid": "1",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "111122223333",
            "444455556666"
         ]
      },
      "Action": [
         "sqs:SendMessage",
         "sqs:ReceiveMessage"
      ],
      "Resource": "arn:aws:sqs:us-east-2:444455556666:queue2",
      "Condition": {
         "DateLessThan": {
```

```
            "AWS:CurrentTime": "2009-06-30T12:00Z"
         }
      }
   }]
}
```

Exemple 3 : Accorder une autorisation à des demandes provenant d'instances Amazon EC2

L'exemple de stratégie Amazon SQS suivant donne accès aux demandes provenant d'instances Amazon EC2. Cet exemple repose sur l'exemple « Exemple 2 : Accorder une autorisation à un ou plusieurs comptes » : il restreint l'accès aux demandes envoyées avant le 30 juin 2009 à midi (UTC), ainsi qu'à la plage d'adresses IP `203.0.113.0/24`. Il est nécessaire d'écrire cette stratégie et de la télécharger dans Amazon SQS à l'aide de l'action `SetQueueAttributes`, car l'action `AddPermission` ne permet pas de spécifier une restriction d'adresse IP lors de l'octroi de l'accès à une file d'attente.

```
{
   "Version": "2012-10-17",
   "Id": "UseCase3",
   "Statement" : [{
      "Sid": "1",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "111122223333"
         ]
      },
      "Action": [
         "sqs:SendMessage",
         "sqs:ReceiveMessage"
      ],
      "Resource": "arn:aws:sqs:us-east-2:444455556666:queue2",
      "Condition": {
         "DateLessThan": {
            "AWS:CurrentTime": "2009-06-30T12:00Z"
         },
         "IpAddress": {
            "AWS:SourceIp": "203.0.113.0/24"
         }
      }
   }]
}
```

Exemple 4 : Refus d'accès à un compte spécifique

L'exemple suivant de politique Amazon SQS refuse un Compte AWS accès spécifique à votre file d'attente. Cet exemple s'appuie sur l'exemple Exemple 1 : Accorder une autorisation à un compte « » : il refuse l'accès au spécifié Compte AWS. Il est nécessaire d'écrire cette stratégie et de la télécharger dans Amazon SQS à l'aide de l'action `SetQueueAttributes`, car l'action `AddPermission` ne permet pas de refuser l'accès à une file d'attente (elle permet uniquement d'accorder l'accès à une file d'attente).

```
{
   "Version": "2012-10-17",
   "Id": "UseCase4",
   "Statement" : [{
      "Sid": "1",
      "Effect": "Deny",
      "Principal": {
         "AWS": [
            "111122223333"
         ]
      },
      "Action": [
         "sqs:SendMessage",
         "sqs:ReceiveMessage"
      ],
      "Resource": "arn:aws:sqs:us-east-2:444455556666:queue2"
   }]
}
```

Exemple 5 : Refuser l'accès s'il n'émane pas d'un point de terminaison de VPC

L'exemple suivant de stratégie Amazon SQS restreint l'accès à `queue1` : 111122223333 peut effectuer les actions `SendMessage` et `ReceiveMessage` uniquement à partir de l'ID de point de terminaison d'un VPC `vpce-1a2b3c4d` (spécifié à l'aide de la condition `aws:sourceVpce`). Pour plus d'informations, consultez Points de terminaison Amazon Virtual Private Cloud pour Amazon SQS.

> **Note**
>
> - La condition `aws:sourceVpce` ne requiert pas d'ARN pour la ressource du point de terminaison de VPC, uniquement l'ID du point de terminaison de VPC.

- Vous pouvez modifier l'exemple suivant pour restreindre toutes les actions au point de terminaison d'un VPC spécifique en refusant toutes les actions Amazon SQS (`sqs:*`) dans la deuxième instruction. Toutefois, une telle déclaration de stratégie stipulerait que toutes les actions (y compris les actions administratives requises pour modifier les autorisations de la file d'attente) doivent être effectuées via le point de terminaison de VPC spécifique défini dans la stratégie, ce qui pourrait empêcher l'utilisateur de modifier les autorisations de la file d'attente par la suite.

```
{
   "Version": "2012-10-17",
   "Id": "UseCase5",
   "Statement": [{
      "Sid": "1",
      "Effect": "Allow",
      "Principal": {
         "AWS": [
            "111122223333"
         ]
      },
      "Action": [
         "sqs:SendMessage",
         "sqs:ReceiveMessage"
      ],
         "Resource": "arn:aws:sqs:us-east-2:111122223333:queue1"
      },
      {
         "Sid": "2",
         "Effect": "Deny",
         "Principal": "*",
         "Action": [
            "sqs:SendMessage",
            "sqs:ReceiveMessage"
         ],
         "Resource": "arn:aws:sqs:us-east-2:111122223333:queue1",
         "Condition": {
            "StringNotEquals": {
               "aws:sourceVpce": "vpce-1a2b3c4d"
            }
         }
      }
```

```
    ]
}
```

Utilisation d'informations d'identification de sécurité temporaires avec Amazon SQS

En plus de créer des utilisateurs dotés de leurs propres identifiants de sécurité, IAM vous permet également d'octroyer des identifiants de sécurité temporaires à n'importe quel utilisateur, lui permettant ainsi d'accéder à vos AWS services et ressources. Vous pouvez gérer les utilisateurs qui possèdent des Comptes AWS. Vous pouvez également gérer les utilisateurs de votre système qui n'en ont pas Comptes AWS (utilisateurs fédérés). En outre, les applications que vous créez pour accéder à vos AWS ressources peuvent également être considérées comme des « utilisateurs ».

Vous pouvez utiliser ces informations d'identification de sécurité temporaires pour effectuer des demandes à Amazon SQS. Les bibliothèques d'API calculent la valeur de signature nécessaire en utilisant ces informations d'identification pour authentifier votre demande. Si vous envoyez des demandes en utilisant des informations d'identification expirées, Amazon SQS les rejette.

> **Note**
>
> Vous ne pouvez pas définir une stratégie en fonction d'informations d'identification temporaires.

Prérequis

1. Utilisez IAM pour créer des informations d'identification de sécurité temporaires :
   - Jeton de sécurité
   - ID de clé d'accès
   - Clé d'accès secrète
2. Préparez votre chaîne pour signer avec l'ID de clé d'accès temporaire et le jeton de sécurité.
3. Utilisez la clé d'accès secrète temporaire au lieu de votre propre clé d'accès secrète pour signer votre demande d'API de requête.

> **Note**
>
> Lorsque vous envoyez la demande d'API de requête signée, utilisez l'ID de clé d'accès temporaire au lieu de votre propre ID de clé d'accès, et incluez le jeton de sécurité. Pour plus

d'informations sur la prise en charge par IAM des informations d'identification de sécurité temporaires, consultez la section Octroi d'un accès temporaire à vos AWS ressources dans le guide de l'utilisateur IAM.

Pour appeler une action d'API de requête Amazon SQS à l'aide d'informations d'identification de sécurité temporaires

1. Demandez un jeton de sécurité temporaire en utilisant AWS Identity and Access Management. Pour plus d'informations, consultez Création d'informations d'identification de sécurité temporaires pour activer l'accès pour les utilisateurs IAM dans le Guide de l'utilisateur IAM.

   IAM renvoie un jeton de sécurité, un ID de clé d'accès et une clé d'accès secrète.
2. Préparez votre requête en utilisant l'ID de clé d'accès temporaire à la place de votre propre ID de clé d'accès, et incluez le jeton de sécurité. Signez votre demande à l'aide de la clé d'accès secrète temporaire au lieu de la vôtre.
3. Soumettez votre chaîne de requête signée avec l'ID de clé d'accès temporaire et le jeton de sécurité.

   L'exemple suivant montre comment utiliser des informations d'identification de sécurité temporaires pour authentifier une demande Amazon SQS. La structure de *AUTHPARAMS* dépend de la signature de la demande d'API. Pour plus d'informations, consultez Signing AWS API Requests dans le manuel Amazon Web Services General Reference.

   ```
   https://sqs.us-east-2.amazonaws.com/
   ?Action=CreateQueue
   &DefaultVisibilityTimeout=40
   &QueueName=MyQueue
   &Attribute.1.Name=VisibilityTimeout
   &Attribute.1.Value=40
   &Expires=2020-12-18T22%3A52%3A43PST
   &SecurityToken=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
   &AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE
   &Version=2012-11-05
   &AUTHPARAMS
   ```

   L'exemple suivant utilise des informations d'identification de sécurité temporaires pour envoyer deux messages à l'aide de l'action `SendMessageBatch`.

```
https://sqs.us-east-2.amazonaws.com/
?Action=SendMessageBatch
&SendMessageBatchRequestEntry.1.Id=test_msg_001
&SendMessageBatchRequestEntry.1.MessageBody=test%20message%20body%201
&SendMessageBatchRequestEntry.2.Id=test_msg_002
&SendMessageBatchRequestEntry.2.MessageBody=test%20message%20body%202
&SendMessageBatchRequestEntry.2.DelaySeconds=60
&Expires=2020-12-18T22%3A52%3A43PST
&SecurityToken=je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
&AWSAccessKeyId=AKIAI44QH8DHBEXAMPLE
&Version=2012-11-05
&AUTHPARAMS
```

Gestion de l'accès pour les files d'attente Amazon SQS chiffrées avec des politiques de moindre privilège

Vous pouvez utiliser Amazon SQS pour échanger des données sensibles entre les applications à l'aide du chiffrement côté serveur (SSE) intégré àAWS Key Management Service (KMS). Grâce à l'intégration d'Amazon SQS AWS KMS, vous pouvez gérer de manière centralisée les clés qui protègent Amazon SQS, ainsi que les clés qui protègent vos autres ressources. AWS

Plusieurs AWS services peuvent agir comme des sources d'événements qui envoient des événements à Amazon SQS. Pour permettre à une source d'événements d'accéder à la file d'attente cryptée Amazon SQS, vous devez configurer la file d'attente avec une clé gérée par le client AWS KMS . Utilisez ensuite la politique clé pour autoriser le service à utiliser les méthodes d' AWS KMS API requises. Le service requiert également des autorisations pour authentifier l'accès et permettre à la file d'attente d'envoyer des événements. Pour cela, vous pouvez utiliser une stratégie Amazon SQS, qui est basée sur les ressources que vous pouvez utiliser pour contrôler l'accès à la file d'attente Amazon SQS et à ses données.

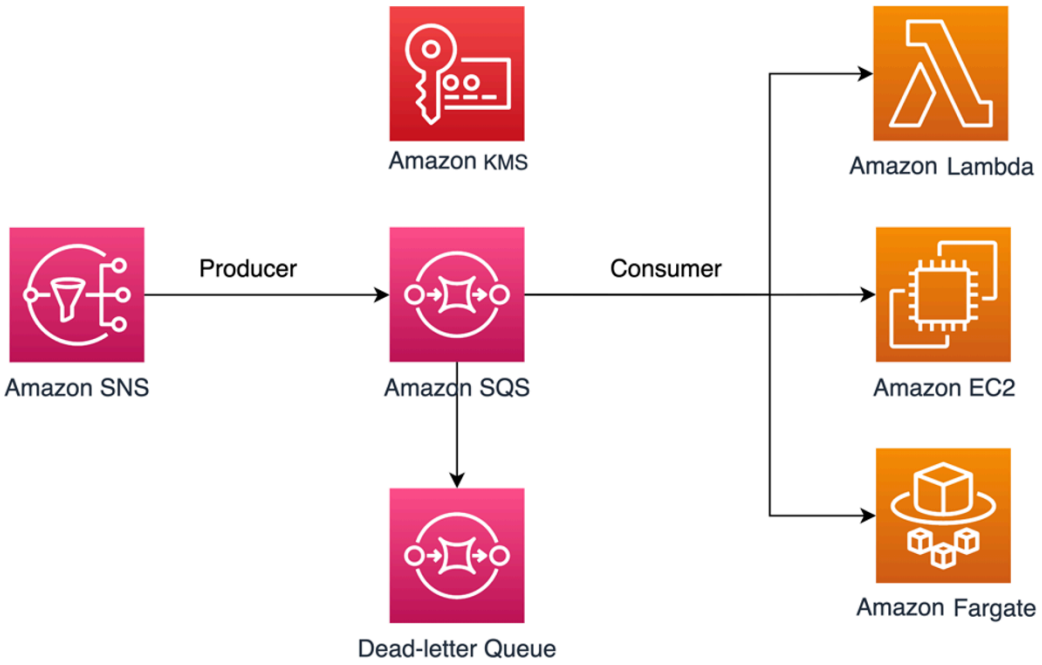
Les sections suivantes fournissent des informations sur la façon de contrôler l'accès à votre file d'attente Amazon SQS chiffrée par le biais de la politique Amazon SQS et AWS KMS de la politique clé. Les stratégies décrites dans ce guide vous aideront à respecter le principe de moindre privilège.

Ce guide décrit également comment les stratégies basées sur les ressources résolvent le problème de député confus en utilisant les clés contextuelles de condition IAM globales `aws:SourceArn`, `aws:SourceAccount` et `aws:PrincipalOrgID`.

Rubriques

Présentation

Dans cette rubrique, nous allons vous présenter un cas d'utilisation courant pour illustrer comment créer la stratégie de clé et la stratégie de file d'attente Amazon SQS. Ce cas d'utilisation est représenté dans l'image suivante.



Dans cet exemple, le producteur du message est une rubrique Amazon Simple Notification Service (SNS) configurée pour diffuser en éventail les messages vers votre file d'attente Amazon SQS chiffrée. Le consommateur de messages est un service informatique, tel qu'une fonction AWS Lambda, une instance Amazon Elastic Compute Cloud (EC2) ou un conteneur AWS Fargate. Votre file d'attente Amazon SQS est ensuite configurée pour envoyer les messages en échec à une file d'attente de lettres mortes (DLQ). Cela est utile pour le débogage de votre application ou de votre système de messagerie, car les DLQ vous permettent d'isoler les messages non consommés afin de déterminer pourquoi leur traitement a échoué. Dans la solution définie dans cette rubrique, un service de calcul tel qu'une fonction Lambda est utilisé pour traiter les messages stockés dans la file d'attente Amazon SQS. Si le destinataire du message se trouve dans un cloud privé virtuel (VPC), l'instruction

de la stratégie `DenyReceivingIfNotThroughVPCE` incluse dans ce guide vous permet de limiter la réception des messages à ce VPC spécifique.

> **Note**
>
> Ce guide contient uniquement les autorisations IAM requises sous forme d'instructions de stratégie. Pour élaborer la politique, vous devez ajouter les instructions à votre politique Amazon SQS ou à votre politique AWS KMS clé. Ce guide ne fournit pas d'instructions sur la façon de créer la file d'attente Amazon SQS ou la AWS KMS clé. Pour savoir comment créer ces ressources, consultez les sections Création d'une file d'attente Amazon SQS et Création de clés.
>
> La stratégie Amazon SQS définie dans ce guide ne permet pas de rediriger les messages directement vers la même file d'attente Amazon SQS ou vers une autre.

Stratégie de clé respectant le principe du moindre privilège pour Amazon SQS

Dans cette section, nous décrivons les autorisations de moindre privilège requises AWS KMS pour la clé gérée par le client que vous utilisez pour chiffrer votre file d'attente Amazon SQS. Avec ces autorisations, vous pouvez limiter l'accès aux seules entités prévues en implémentant le moindre privilège. La stratégie de clé doit comprendre les instructions de stratégie suivantes, que nous décrivons en détail ci-dessous :

- Accorder des autorisations d'administrateur à la AWS KMS clé
- Accorde l'accès en lecture seule aux métadonnées de clés
- Accorder des autorisations KMS Amazon SNS à Amazon SNS pour la publication de messages dans la file d'attente
- Permettre aux consommateurs de déchiffrer les messages de la file d'attente

Accorder des autorisations d'administrateur à la AWS KMS clé

Pour créer une AWS KMS clé, vous devez fournir des autorisations d' AWS KMS administrateur au rôle IAM que vous utilisez pour déployer la AWS KMS clé. Ces autorisations d'administrateur sont définies dans l'instruction de stratégie `AllowKeyAdminPermissions` suivante. Lorsque vous ajoutez cette déclaration à votre politique AWS KMS clé, assurez-vous de <admin-role ARN>la remplacer par le nom de ressource Amazon (ARN) du rôle IAM utilisé pour déployer la AWS KMS clé,

gérer la AWS KMS clé, ou les deux. Il peut s'agir du rôle IAM de votre pipeline de déploiement ou du rôle d'administrateur de votre organisation dans AWS Organizations.

```
{
  "Sid": "AllowKeyAdminPermissions",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
      "<admin-role ARN>"
    ]
  },
  "Action": [
    "kms:Create*",
    "kms:Describe*",
    "kms:Enable*",
    "kms:List*",
    "kms:Put*",
    "kms:Update*",
    "kms:Revoke*",
    "kms:Disable*",
    "kms:Get*",
    "kms:Delete*",
    "kms:TagResource",
    "kms:UntagResource",
    "kms:ScheduleKeyDeletion",
    "kms:CancelKeyDeletion"
  ],
  "Resource": "*"
}
```

> **Note**
>
> Dans une politique AWS KMS clé, la valeur de l'`Resource`élément doit être`*`, ce qui signifie « cette AWS KMS clé ». L'astérisque (`*`) identifie la AWS KMS clé à laquelle la politique clé est attachée.

Accorde l'accès en lecture seule aux métadonnées de clés

Pour accorder à d'autres rôles IAM un accès en lecture seule à vos métadonnées de clés, ajoutez l'instruction `AllowReadAccessToKeyMetaData` à votre stratégie de clé. Par exemple, l'instruction

suivante vous permet de répertorier toutes les AWS KMS clés de votre compte à des fins d'audit. Cette instruction accorde à l'utilisateur AWS root un accès en lecture seule aux métadonnées clés. Par conséquent, tout principal IAM du compte peut avoir accès aux métadonnées de clés lorsque ses stratégies basées sur l'identité disposent des autorisations répertoriées dans l'instruction suivante : `kms:Describe*`, `kms:Get*` et `kms:List*`. Veillez à remplacer *<account-ID>* par vos propres informations.

```
{
  "Sid": "AllowReadAcesssToKeyMetaData",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
      "arn:aws:iam::<accountID>:root"
    ]
  },
  "Action": [
    "kms:Describe*",
    "kms:Get*",
    "kms:List*"
  ],
  "Resource": "*"
}
```

Accorder des autorisations KMS Amazon SNS à Amazon SNS pour la publication de messages dans la file d'attente

Pour permettre à votre rubrique Amazon SNS de publier des messages dans votre file d'attente Amazon SQS chiffrée, ajoutez l'instruction de stratégie `AllowSNSToSendToSQS` à votre stratégie de clé. Cette déclaration autorise Amazon SNS à utiliser la AWS KMS clé pour publier dans votre file d'attente Amazon SQS. Veillez à remplacer *<account-ID>* par vos propres informations.

> **Note**
>
> La déclaration `Condition` in the limite l'accès au seul service Amazon SNS sur le même AWS compte.

```
{
  "Sid": "AllowSNSToSendToSQS",
  "Effect": "Allow",
```

```
  "Principal": {
    "Service": [
      "sns.amazonaws.com"
    ]
  },
  "Action": [
    "kms:GenerateDataKey",
    "kms:Decrypt"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "aws:SourceAccount": "<account-id>"
    }
  }
}
```

Permettre aux consommateurs de déchiffrer les messages de la file d'attente

L'instruction `AllowConsumersToReceiveFromTheQueue` suivante accorde au consommateur de messages Amazon SQS les autorisations requises pour déchiffrer les messages reçus de la file d'attente Amazon SQS chiffrée. Lorsque vous joignez l'instruction de stratégie, remplacez *<consumer's runtime role ARN>* par l'ARN du rôle d'exécution IAM du consommateur de messages.

```
{
  "Sid": "AllowConsumersToReceiveFromTheQueue",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
      "<consumer's execution role ARN>"
    ]
  },
  "Action": [
    "kms:Decrypt"
  ],
  "Resource": "*"
}
```

Stratégie Amazon SQS du moindre privilège

Cette section décrit les stratégies de file d'attente Amazon SQS relatives au moindre privilège pour le cas d'utilisation couvert par ce guide (par exemple, Amazon SNS vers Amazon SQS). La stratégie définie est conçue pour empêcher tout accès involontaire en utilisant à la fois des instructions `Deny` et `Allow`. Les instructions `Allow` donnent accès à l'entité ou aux entités prévues. Les instructions `Deny` évitent que d'autres entités indésirables n'accèdent à la file d'attente Amazon SQS, tout en excluant l'entité prévue dans la condition de stratégie.

La stratégie Amazon SQS inclut les instructions suivantes, que nous décrivons en détail ci-dessous :

- Restreindre les autorisations de gestion Amazon SQS
- Restreindre les actions de file d'attente Amazon SQS pour l'organisation spécifiée
- Accorder des autorisations Amazon SQS aux consommateurs
- Application du chiffrement en transit
- Limiter la transmission de messages à une rubrique Amazon SNS spécifique
- (Facultatif) Restreindre la réception des messages au point de terminaison d'un VPC spécifique

Restreindre les autorisations de gestion Amazon SQS

L'instruction de stratégie `RestrictAdminQueueActions` suivante limite les autorisations de gestion Amazon SQS uniquement au(x) rôle(s) IAM que vous utilisez pour déployer la file d'attente, la gérer, ou les deux. Assurez-vous de remplacer les *<valeurs d'espace réservé>* par vos propres informations. Spécifiez l'ARN du rôle IAM utilisé pour déployer la file d'attente Amazon SQS, ainsi que les ARN de tous les rôles d'administrateur devant disposer d'autorisations de gestion Amazon SQS.

```
{
  "Sid": "RestrictAdminQueueActions",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": [
    "sqs:AddPermission",
    "sqs:DeleteQueue",
    "sqs:RemovePermission",
    "sqs:SetQueueAttributes"
  ],
```

```
  "Resource": "<SQS Queue ARN>",
  "Condition": {
    "StringNotLike": {
      "aws:PrincipalARN": [
        "arn:aws:iam::<account-id>:role/<deployment-role-name>",
        "<admin-role ARN>"
      ]
    }
  }
}
```

Restreindre les actions de file d'attente Amazon SQS pour l'organisation spécifiée

Pour protéger vos ressources Amazon SQS contre tout accès externe (accès par une entité extérieure à votre organisation AWS), utilisez l'instruction suivante. Cette instruction limite l'accès à la file d'attente Amazon SQS pour l'organisation que vous spécifiez dans la `Condition`. Assurez-vous de remplacer *<SQS queue ARN>* par l'ARN du rôle IAM utilisé pour déployer la file d'attente Amazon SQS, et *<org-id>* par l'ID de votre organisation.

```
{
  "Sid": "DenyQueueActionsOutsideOrg",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": [
    "sqs:AddPermission",
    "sqs:ChangeMessageVisibility",
    "sqs:DeleteQueue",
    "sqs:RemovePermission",
    "sqs:SetQueueAttributes",
    "sqs:ReceiveMessage"
  ],
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "StringNotEquals": {
      "aws:PrincipalOrgID": [
        "<org-id>"
      ]
    }
  }
}
```

Accorder des autorisations Amazon SQS aux consommateurs

Pour recevoir des messages de la file d'attente Amazon SQS, vous devez fournir les autorisations nécessaires au consommateur des messages. L'instruction de stratégie suivante accorde au consommateur spécifié les autorisations requises pour consommer les messages de la file d'attente Amazon SQS. Lorsque vous ajoutez cette instruction à votre stratégie Amazon SQS, assurez-vous de remplacer *<consumer's IAM runtime role ARN>* par l'ARN du rôle d'exécution IAM utilisé par le consommateur, et *<SQS queue ARN>* par l'ARN du rôle IAM utilisé pour déployer la file d'attente Amazon SQS.

```
{
  "Sid": "AllowConsumersToReceiveFromTheQueue",
  "Effect": "Allow",
  "Principal": {
    "AWS": "<consumer's IAM execution role ARN>"
  },
  "Action": [
    "sqs:ChangeMessageVisibility",
    "sqs:DeleteMessage",
    "sqs:GetQueueAttributes",
    "sqs:ReceiveMessage"
  ],
  "Resource": "<SQS queue ARN>"
}
```

Pour éviter que d'autres entités ne reçoivent des messages de la file d'attente Amazon SQS, ajoutez l'instruction `DenyOtherConsumersFromReceiving` à la stratégie de file d'attente Amazon SQS. Cette instruction limite la consommation des messages au consommateur spécifié et n'autorise aucun autre consommateur à y avoir accès, même lorsque ses autorisations d'identité l'y autorisent. Assurez-vous de remplacer *<SQS queue ARN>* et *<consumer's runtime role ARN>* par vos propres informations.

```
{
  "Sid": "DenyOtherConsumersFromReceiving",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": [
```

```
    "sqs:ChangeMessageVisibility",
    "sqs:DeleteMessage",
    "sqs:ReceiveMessage"
  ],
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "StringNotLike": {
      "aws:PrincipalARN": "<consumer's execution role ARN>"
    }
  }
}
```

Application du chiffrement en transit

L'instruction de stratégie `DenyUnsecureTransport` suivante oblige les consommateurs et les producteurs à utiliser des canaux sécurisés (connexions TLS) pour envoyer et recevoir des messages depuis la file d'attente Amazon SQS. Assurez-vous de remplacer *<SQS queue ARN>* par l'ARN du rôle IAM utilisé pour déployer la file d'attente Amazon SQS.

```
{
  "Sid": "DenyUnsecureTransport",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": [
    "sqs:ReceiveMessage",
    "sqs:SendMessage"
  ],
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "Bool": {
      "aws:SecureTransport": "false"
    }
  }
}
```

Limiter la transmission de messages à une rubrique Amazon SNS spécifique

L'instruction de stratégie `AllowSNSToSendToTheQueue` suivante permet à la rubrique Amazon SNS spécifiée d'envoyer des messages à la file d'attente Amazon SQS. Assurez-vous de remplacer *<SQS queue ARN>* par l'ARN du rôle IAM utilisé pour déployer la file d'attente Amazon SQS, et *<SNS topic ARN>* par l'ARN de la rubrique Amazon SNS.

```
{
  "Sid": "AllowSNSToSendToTheQueue",
  "Effect": "Allow",
  "Principal": {
    "Service": "sns.amazonaws.com"
  },
  "Action": "sqs:SendMessage",
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "ArnLike": {
      "aws:SourceArn": "<SNS topic ARN>"
    }
  }
}
```

L'instruction de stratégie `DenyAllProducersExceptSNSFromSending` suivante empêche les autres producteurs d'envoyer des messages à la file d'attente. Remplacez *<SQS queue ARN>* et *<SNS topic ARN>* par vos propres informations.

```
{
  "Sid": "DenyAllProducersExceptSNSFromSending",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": "sqs:SendMessage",
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "ArnNotLike": {
      "aws:SourceArn": "<SNS topic ARN>"
    }
  }
}
```

(Facultatif) Restreindre la réception des messages au point de terminaison d'un VPC spécifique

Pour limiter la réception de messages au point de terminaison d'un VPC spécifique, ajoutez l'instruction de stratégie suivante à votre stratégie de file d'attente Amazon SQS. Cette instruction empêche un consommateur de messages de recevoir des messages de la file d'attente, sauf si les messages proviennent du point de terminaison d'un VPC souhaité. Remplacez *<SQS queue ARN>* par l'ARN du rôle IAM utilisé pour déployer la file d'attente Amazon SQS et *<vpce_id>* par l'ID du point de terminaison d'un VPC.

```
{
  "Sid": "DenyReceivingIfNotThroughVPCE",
  "Effect": "Deny",
  "Principal": "*",
  "Action": [
    "sqs:ReceiveMessage"
  ],
  "Resource": "<SQS queue ARN>",
  "Condition": {
    "StringNotEquals": {
      "aws:sourceVpce": "<vpce id>"
    }
  }
}
```

Instructions de stratégie Amazon SQS relatives à la file d'attente de lettres mortes

Ajoutez les instructions de stratégie suivantes, identifiées par leur ID d'instruction, à votre stratégie d'accès DLQ :

- `RestrictAdminQueueActions`
- `DenyQueueActionsOutsideOrg`
- `AllowConsumersToReceiveFromTheQueue`
- `DenyOtherConsumersFromReceiving`
- `DenyUnsecureTransport`

Outre l'ajout des instructions de stratégie précédentes à votre stratégie d'accès DLQ, vous devez également ajouter une instruction pour restreindre la transmission de messages aux files d'attente Amazon SQS, comme décrit dans la section suivante.

Restreindre la transmission de messages vers des files d'attente Amazon SQS

Pour restreindre l'accès aux files d'attente Amazon SQS provenant du même compte, ajoutez l'instruction de stratégie `DenyAnyProducersExceptSQS` suivante à la stratégie de file d'attente DLQ. Cette instruction ne limite pas la transmission de messages à une file d'attente spécifique, car vous devez déployer le DLQ avant de créer la file d'attente principale. Vous ne connaîtrez donc pas l'ARN Amazon SQS lorsque vous créerez le DLQ. Si vous devez limiter l'accès à une seule file d'attente Amazon SQS, modifiez `aws:SourceArn` dans la `Condition` avec l'ARN de votre file d'attente source Amazon SQS lorsque vous le connaîtrez.

```
{
  "Sid": "DenyAnyProducersExceptSQS",
  "Effect": "Deny",
  "Principal": {
    "AWS": "*"
  },
  "Action": "sqs:SendMessage",
  "Resource": "<SQS DLQ ARN>",
  "Condition": {
    "ArnNotLike": {
      "aws:SourceArn": "arn:aws:sqs:<region>:<account-id>:*"
    }
  }
}
```

Important

Les stratégies de file d'attente Amazon SQS définies dans ce guide ne limitent pas l'action `sqs:PurgeQueue` à un ou plusieurs rôles IAM spécifiques. L'action `sqs:PurgeQueue` vous permet de supprimer tous les messages de la file d'attente Amazon SQS. Vous pouvez également utiliser cette action pour modifier le format du message sans remplacer la file d'attente Amazon SQS. Lors du débogage d'une application, vous pouvez effacer la file d'attente Amazon SQS pour supprimer les messages potentiellement erronés. Lorsque vous testez l'application, vous pouvez générer un volume élevé de messages dans la file d'attente Amazon SQS, puis purger la file d'attente pour repartir à zéro avant de passer à la production. La raison pour laquelle cette action n'est pas limitée à un certain rôle est que

ce rôle peut ne pas être connu lors du déploiement de la file d'attente Amazon SQS. Vous devrez ajouter cette autorisation à la stratégie basée sur l'identité du rôle pour pouvoir purger la file d'attente.

Prévention des problèmes de député confus entre services

Le problème de député confus est un problème de sécurité dans lequel une entité qui n'a pas l'autorisation d'effectuer une action peut contraindre une entité plus privilégiée à effectuer cette action. 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. Les instructions de stratégie de cette section peuvent vous aider à éviter le problème de député confus entre services.

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. Pour éviter ce problème, les stratégies basées sur les ressources définies dans cet article utilisent les clés contextuelles de condition IAM globales `aws:SourceArn`, `aws:SourceAccount` et `aws:PrincipalOrgID`. Cela limite les autorisations dont dispose un service pour une ressource spécifique, un compte spécifique ou une organisation spécifique dans AWS Organizations.

Utiliser IAM Access Analyzer pour examiner l'accès intercompte

Vous pouvez utiliser AWS IAM Access Analyzer pour examiner vos politiques de file d'attente Amazon SQS et vos politiques clés AWS KMS et vous avertir lorsqu'une file d'attente ou AWS KMS une clé Amazon SQS autorise l'accès à une entité externe. IAM Access Analyzer vous aide à identifier les ressources de votre organisation et de vos comptes partagés avec une entité située en dehors de la zone de confiance. Cette zone de confiance peut être un AWS compte ou l'organisation au sein d' AWS Organizations que vous spécifiez lorsque vous activez IAM Access Analyzer.

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 zone de confiance, Access Analyzer génère un résultat. Les résultats comprennent des renseignements sur l'accès et le principal externe à qui il est accordé. Révisez 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 cas d'accès involontaire, consultez la stratégie concernée et corrigez-la. Consultez

ce billet de blog pour plus d'informations sur la manière dont AWS IAM Access Analyzer identifie les accès involontaires à vos ressources. AWS

Pour plus d'informations sur AWS IAM Access Analyzer, consultez la documentation d'AWS IAM Access Analyzer.

Autorisations d'API Amazon SQS : référence des actions et ressources

Vous pouvez utiliser le tableau ci-dessous comme référence lorsque vous configurez le Contrôle d'accès et que vous écrivez des stratégies d'autorisation que vous pouvez associer à une identité IAM. Le chaque action d'Amazon Simple Queue Service, les actions correspondantes pour lesquelles vous pouvez accorder des autorisations pour effectuer l'action et la AWS ressource pour laquelle vous pouvez accorder les autorisations.

Spécifiez les actions dans le champ `Action` de la stratégie, et la valeur des ressources dans le champ `Resource` de la stratégie. Pour spécifier une action, utilisez le préfixe `sqs:` suivi du nom de l'action (par exemple, `sqs:CreateQueue`).

Amazon SQS prend actuellement en charge les clés contextuelles de condition globales disponibles dans IAM.

API Amazon Simple Queue Service et autorisations requises pour les actions

AddPermission

Action(s) : `sqs:AddPermission`

Ressource : `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

ChangeMessageVisibilité

Action(s) : `sqs:ChangeMessageVisibility`

Ressource : `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

ChangeMessageVisibilityBatch

Action(s) : `sqs:ChangeMessageVisibilityBatch`

Ressource : `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

CreateQueue

Action(s) : `sqs:CreateQueue`

Ressource : `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

DeleteMessage

Action(s) : `sqs:DeleteMessage`

Ressource : `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

DeleteMessageBatch

Action(s) : `sqs:DeleteMessageBatch`

Ressource : `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

DeleteQueue

Action(s) : `sqs:DeleteQueue`

Ressource : `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

GetQueueAttributs

Action(s) : `sqs:GetQueueAttributes`

Ressource : `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

GetQueueURL

Action(s) : `sqs:GetQueueUrl`

Ressource : `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

ListDeadLetterSourcefiles d'attente

Action(s) : `sqs:ListDeadLetterSourceQueues`

Ressource : `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

ListQueues

Action(s) : `sqs:ListQueues`

Ressource : `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

ListQueueBalises

Action(s) : `sqs:ListQueueTags`

Ressource : `arn:aws:sqs:`*`region`*`:`*`account_id`*`:`*`queue_name`*

PurgeQueue

Action(s) : `sqs:PurgeQueue`

Ressource : `arn:aws:sqs:region:account_id:queue_name`

ReceiveMessage

Action(s) : `sqs:ReceiveMessage`

Ressource : `arn:aws:sqs:region:account_id:queue_name`

RemovePermission

Action(s) : `sqs:RemovePermission`

Ressource : `arn:aws:sqs:region:account_id:queue_name`

SendMessageet SendMessageBatch

Action(s) : `sqs:SendMessage`

Ressource : `arn:aws:sqs:region:account_id:queue_name`

SetQueueAttributs

Action(s) : `sqs:SetQueueAttributes`

Ressource : `arn:aws:sqs:region:account_id:queue_name`

TagQueue

Action(s) : `sqs:TagQueue`

Ressource : `arn:aws:sqs:region:account_id:queue_name`

UntagQueue

Action(s) : `sqs:UntagQueue`

Ressource : `arn:aws:sqs:region:account_id:queue_name`

Journalisation et surveillance dans Amazon SQS

Cette section fournit des informations sur les options de journalisation et de surveillance pour Amazon SQS, notamment sur la façon de les utiliser CloudTrail pour capturer des appels d'API, ainsi que des CloudWatch indicateurs permettant d'obtenir des informations sur l'activité et les performances des files d'attente.

Rubriques

Journalisation des appels d'API Amazon SQS à l'aide d' AWS CloudTrail

Amazon SQS est intégré AWS CloudTrail pour enregistrer les appels Amazon SQS d'un utilisateur, d'un rôle ou d'un service. AWS CloudTrail capture les appels d'API liés à la norme Amazon SQS et aux files d'attente FIFO sous forme d'événements, y compris les interactions initiées via la console Amazon SQS ou de manière programmatique via des appels aux API Amazon SQS.

Rubriques

Informations Amazon SQS dans CloudTrail

CloudTrail est activé par défaut lorsque vous créez votre AWS compte. Lorsqu'une activité d'événement Amazon SQS prise en charge se produit, elle 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 pour votre AWS compte. Pour plus d'informations, consultez la section Affichage des événements avec l'historique des CloudTrail événements dans le guide de AWS CloudTrail l'utilisateur.

Les API Amazon SQS qui appellent des opérations de gestion des files d'attente, telles que celles-ci, `AddPermission` sont classées comme des événements de gestion et sont connectées CloudTrail par défaut. Les API Amazon SQS qui sont des opérations à volume élevé effectuées sur une file d'attente Amazon SQS, par exemple, sont `SendMessage` classées dans la catégorie des événements de données et sont enregistrées une fois que vous vous êtes inscrit. CloudTrail

À l'aide des informations CloudTrail collectées, vous pouvez identifier une demande spécifique adressée à une API Amazon SQS, l'adresse IP ou l'identité du demandeur, ainsi que la date et l'heure de la demande. Si vous configurez un CloudTrail suivi, vous pouvez diffuser des CloudTrail événements en continu vers un compartiment Amazon S3 avec une diffusion facultative vers Amazon

CloudWatch Logs et AWS EventBridge. Si vous ne configurez pas de suivi, vous pouvez uniquement consulter l'historique des événements de gestion dans les événements de la CloudTrail console. Pour plus d'informations, consultez Présentation de la création d'un journal de suivi dans le Guide de l'utilisateur AWS CloudTrail.

Événements de gestion dans CloudTrail

Voici les actions d'API qu'Amazon SQS journalise sous forme d'événements de gestion :

- `AddPermission`
- `CreateQueue`
- `CancelMessageMoveTask`
- `DeleteQueue`
- `ListMessageMoveTasks`
- `PurgeQueue`
- `RemovePermission`
- `SetQueueAttributes`
- `StartMessageMoveTask`
- `TagQueue`
- `UntagQueue`

Les API Amazon SQS suivantes ne sont pas prises en charge pour CloudTrail la journalisation :

- `GetQueueAttributes`
- `GetQueueUrl`
- `ListDeadLetterSourceQueues`
- `ListQueueTags`
- `ListQueues`

Événements liés aux données dans CloudTrail

Les événements de données fournissent des informations sur les opérations effectuées pour ou au niveau d'une ressource, telles que l'envoi ou la réception d'un message Amazon SQS vers ou depuis une file d'attente Amazon SQS. Les événements de données sont des activités volumineuses qui CloudTrail ne sont pas enregistrées par défaut. Vous pouvez activer la journalisation des actions de

l'API Data Events pour votre file d'attente SQS à l'aide CloudTrail des API. Pour plus d'informations, consultez Journalisation des événements de données dans le Guide de l'utilisateur AWS CloudTrail .

Avec CloudTrail, vous pouvez utiliser des sélecteurs d'événements avancés pour décider quelles activités de l'API Amazon SQS sont enregistrées et enregistrées. Pour journaliser les événements de données Amazon SQS, vous devez inclure le type de ressource `AWS::SQS::Queue`. Une fois cette configuration effectuée, vous pouvez peaufiner vos préférences de journalisation en spécifiant les événements de données à enregistrer, par exemple en utilisant le filtre `eventName` pour suivre les événements `SendMessage`. Pour plus d'informations, consultez `AdvancedEventSelector` dans la Référence d'API AWS CloudTrail .

Événements de données Amazon SQS :

- `SendMessage`
- `SendMessageBatch`
- `ReceiveMessage`
- `DeleteMessage`
- `DeleteMessageBatch`
- `ChangeMessageVisibility`
- `ChangeMessageVisibilityBatch`

Des frais supplémentaires s'appliquent pour les événements de données. Pour plus d'informations, consultez Tarification d'AWS CloudTrail.

Exemples : événements CloudTrail de gestion pour Amazon SQS

Les exemples suivants montrent les entrées du CloudTrail journal pour les API prises en charge :

AddPermission

L'exemple suivant montre une entrée de CloudTrail journal pour un appel d'`AddPermission`API.

```
{
   "Records": [
     {
       "eventVersion": "1.06",
       "userIdentity": {
         "type": "IAMUser",
         "principalId": "AKIAI44QH8DHBEXAMPLE",
```

```
        "arn": "arn:aws:iam::123456789012:user/Alice",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Alice"
      },
      "eventTime": "2018-06-28T22:23:46Z",
      "eventSource": "sqs.amazonaws.com",
      "eventName": "AddPermission",
      "awsRegion": "us-east-2",
      "sourceIPAddress": "203.0.113.0",
      "userAgent": "Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20100101
 Firefox/24.0",
      "requestParameters": {
        "actions": [
          "SendMessage"
        ],
        "AWSAccountIds": [
          "123456789012"
        ],
        "label": "MyLabel",
        "queueUrl": "https://sqs.us-east-2.amazon.com/123456789012/MyQueue"
      },
      "responseElements": null,
      "requestID": "123abcde-f4gh-50ij-klmn-60o789012p30",
      "eventID": "0987g654-32f1-09e8-d765-c4f3fb2109fa"
    }
  ]
}
```

CreateQueue

L'exemple suivant montre une entrée de CloudTrail journal pour un appel d'`CreateQueue`API.

```
{
   "Records": [
     {
       "eventVersion": "1.06",
       "userIdentity": {
         "type": "IAMUser",
         "principalId": "AKIAI44QH8DHBEXAMPLE",
         "arn": "arn:aws:iam::123456789012:user/Alejandro",
         "accountId": "123456789012",
         "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
         "userName": "Alejandro"
```

```
      },
      "eventTime": "2018-06-28T22:23:46Z",
      "eventSource": "sqs.amazonaws.com",
      "eventName": "CreateQueue",
      "awsRegion": "us-east-2",
      "sourceIPAddress": "203.0.113.1",
      "userAgent": "Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20100101
 Firefox/24.0",
      "requestParameters": {
        "queueName": "MyQueue"
      },
      "responseElements": {
        "queueUrl": "https://sqs.us-east-2.amazon.com/123456789012/MyQueue"
      },
      "requestID": "123abcde-f4gh-50ij-klmn-60o789012p30",
      "eventID": "0987g654-32f1-09e8-d765-c4f3fb2109fa"
    }
  ]
}
```

DeleteQueue

L'exemple suivant montre une entrée de CloudTrail journal pour un appel d'`DeleteQueue`API.

```
{
   "Records": [
     {
       "eventVersion": "1.06",
       "userIdentity": {
         "type": "IAMUser",
         "principalId": "AKIAI44QH8DHBEXAMPLE",
         "arn": "arn:aws:iam::123456789012:user/Carlos",
         "accountId": "123456789012",
         "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
         "userName": "Carlos"
       },
       "eventTime": "2018-06-28T22:23:46Z",
       "eventSource": "sqs.amazonaws.com",
       "eventName": "DeleteQueue",
       "awsRegion": "us-east-2",
       "sourceIPAddress": "203.0.113.2",
       "userAgent": "Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20100101
 Firefox/24.0",
       "requestParameters": {
```

```
        "queueUrl": "https://sqs.us-east-2.amazon.com/123456789012/MyQueue"
      },
      "responseElements": null,
      "requestID": "123abcde-f4gh-50ij-klmn-60o789012p30",
      "eventID": "0987g654-32f1-09e8-d765-c4f3fb2109fa"
    }
  ]
}
```

RemovePermission

L'exemple suivant montre une entrée de CloudTrail journal pour un appel d'`RemovePermission`API.

```
{
   "Records": [
     {
       "eventVersion": "1.06",
       "userIdentity": {
         "type": "IAMUser",
         "principalId": "AKIAI44QH8DHBEXAMPLE",
         "arn": "arn:aws:iam::123456789012:user/Jane",
         "accountId": "123456789012",
         "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
         "userName": "Jane"
       },
       "eventTime": "2018-06-28T22:23:46Z",
       "eventSource": "sqs.amazonaws.com",
       "eventName": "RemovePermission",
       "awsRegion": "us-east-2",
       "sourceIPAddress": "203.0.113.3",
       "userAgent": "Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20100101
 Firefox/24.0",
       "requestParameters": {
         "label": "label",
         "queueUrl": "https://sqs.us-east-2.amazon.com/123456789012/MyQueue"
       },
       "responseElements": null,
       "requestID": "123abcde-f4gh-50ij-klmn-60o789012p30",
       "eventID": "0987g654-32f1-09e8-d765-c4f3fb2109fa"
     }
   ]
 }
```

SetQueueAttributes

L'exemple suivant montre une entrée de CloudTrail journal pour `SetQueueAttributes` :


```
{
   "Records": [
     {
       "eventVersion": "1.06",
       "userIdentity": {
         "type": "IAMUser",
         "principalId": "AKIAI44QH8DHBEXAMPLE",
         "arn": "arn:aws:iam::123456789012:user/Maria",
         "accountId": "123456789012",
         "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
         "userName": "Maria"
       },
       "eventTime": "2018-06-28T22:23:46Z",
       "eventSource": "sqs.amazonaws.com",
       "eventName": "SetQueueAttributes",
       "awsRegion": "us-east-2",
       "sourceIPAddress": "203.0.113.4",
       "userAgent": "Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20100101
 Firefox/24.0",
       "requestParameters": {
         "attributes": {
           "VisibilityTimeout": "100"
         },
         "queueUrl": "https://sqs.us-east-2.amazon.com/123456789012/MyQueue"
       },
       "responseElements": null,
       "requestID": "123abcde-f4gh-50ij-klmn-60o789012p30",
       "eventID": "0987g654-32f1-09e8-d765-c4f3fb2109fa"
     }
   ]
 }
```


Exemples : événements CloudTrail liés aux données pour Amazon SQS

Voici des exemples d' CloudTrail événements spécifiques aux API d'événements de données Amazon SQS :

SendMessage

L'exemple suivant montre un événement CloudTrail de données pour`SendMessage`.


```
    {
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/SessionName",
    "accountId": "123456789012",
    "accessKeyId": "ACCESS_KEY_ID",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed",
        "accountId": "123456789012",
        "userName": "RoleToBeAssumed"
      },
      "attributes": {
        "creationDate": "2023-11-07T22:13:06Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-11-07T23:59:11Z",
  "eventSource": "sqs.amazonaws.com",
  "eventName": "SendMessage",
  "awsRegion": "ap-southeast-4",
  "sourceIPAddress": "10.0.118.80",
  "userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
  "requestParameters": {
    "queueUrl": "https://sqs.ap-southeast-4.amazonaws.com/123456789012/MyQueue",
    "messageBody": "HIDDEN_DUE_TO_SECURITY_REASONS",
    "messageDeduplicationId": "MsgDedupIdSdk1ae1958f2-bbe8-4442-83e7-4916e3b035aa",
    "messageGroupId": "MsgGroupIdSdk16"
  },
  "responseElements": {
    "mD5OfMessageBody": "9a4e3f7a614d9dd9f8722092dbda17a2",
    "mD5OfMessageSystemAttributes": "f88f0587f951b7f5551f18ae699c3a9d",
```

```
    "messageId": "93bb6e2d-1090-416c-81b0-31eb1faa8cd8",
    "sequenceNumber": "18881790870905840128"
  },
  "requestID": "c4584600-fe8a-5aa3-a5ba-1bc42f055fae",
  "eventID": "98c735d8-70e0-4644-9432-b6ced4d791b1",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::SQS::Queue",
      "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "123456789012",
  "eventCategory": "Data",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
  }
```


ReceiveMessage

L'exemple suivant montre un événement CloudTrail de données pour`ReceiveMessage`.


```
    {
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/SessionName",
    "accountId": "123456789012",
    "accessKeyId": "ACCESS_KEY_ID",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed",
        "accountId": "123456789012",
        "userName": "RoleToBeAssumed"
```

```
      },
      "attributes": {
        "creationDate": "2023-11-07T22:13:06Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-11-07T23:59:24Z",
  "eventSource": "sqs.amazonaws.com",
  "eventName": "ReceiveMessage",
  "awsRegion": "ap-southeast-4",
  "sourceIPAddress": "10.0.118.80",
  "userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
  "requestParameters": {
    "queueUrl": "https://sqs.ap-southeast-4.amazonaws.com/123456789012/MyQueue",
    "maxNumberOfMessages": 10
  },
  "responseElements": null,
  "requestID": "8b4d4643-8f49-52cd-a6e8-1b875ed54b99",
  "eventID": "f3f23ab7-b0a4-4b71-afc0-141209c49206",
  "readOnly": true,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::SQS::Queue",
      "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "123456789012",
  "eventCategory": "Data",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
  }
}
```

DeleteMessageBatch

L'exemple suivant montre un événement CloudTrail de données pour`DeleteMessageBatch`.


```
    {
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/SessionName",
    "accountId": "123456789012",
    "accessKeyId": "ACCESS_KEY_ID",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed",
        "accountId": "123456789012",
        "userName": "RoleToBeAssumed"
      },
      "attributes": {
        "creationDate": "2023-11-07T22:13:06Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-11-07T23:59:24Z",
  "eventSource": "sqs.amazonaws.com",
  "eventName": "DeleteMessageBatch",
  "awsRegion": "ap-southeast-4",
  "sourceIPAddress": "10.0.118.80",
  "userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
  "requestParameters": {
    "queueUrl": "https://sqs.ap-southeast-4.amazonaws.com/123456789012/MyQueue",
    "entries": [
      {
        "id": "0",
        "receiptHandle": "AQEBefxM1O4zyZGF87DehbRbmri91w2W7mMdD0GrBjQa8e/
hpb4RbXHPZ9tLBVleECbChQIE5NtaDuoZhZPOkTy0eN46EyRR4jXDzE3AlkbPlX1mA9f2fUuTrXx8aeCoCA3I3woNg3f
hlLS94tjAZqV2krc4BaC2pYgjyHWcW019HwIV8T/bjNMIeZoQwOM5V
+o9vHPfewz5QGr5SKpDo7uE7Umyk5n5CJZvcn1efp/
```

```
mrwtaCIb9M7cCQUYcZm2ZmZDnIO9XpGTAi3m2dQ0M83pnNh0nvDfpkHpoa+hX1TrUmxCupCWHJwA8HFJ1O/
CCJsodMNFthLBA9S57dkBZCsw41G8jAmgQ0MkvZ0UL5mg0OFQQd1Yrw0zvthjCgiwdzn0yXoMzxIZMBxkY14E4nVVZ7N
h8oRk2C7gByzg2kYJ0LnUvLJFT8DQE28JZppEC9klvrdR/BWiPT7asc="
      }
    ]
  },
  "responseElements": {
    "successful": [
      {
        "id": "0"
      }
    ],
    "failed": []
  },
  "requestID": "fe423091-5642-5ba5-9256-6d5587de52f1",
  "eventID": "88c8020d-d769-4985-8ecb-ee0b59acc418",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::SQS::Queue",
      "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "123456789012",
  "eventCategory": "Data",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
  }
}
```


ChangeMessageVisibilityBatch

L'exemple suivant montre un événement CloudTrail de données pour`ChangeMessageVisibilityBatch`.


```
  {
```

```
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "EXAMPLE_PRINCIPAL_ID",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/SessionName",
    "accountId": "123456789012",
    "accessKeyId": "ACCESS_KEY_ID",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed",
        "accountId": "123456789012",
        "userName": "RoleToBeAssumed"
      },
      "attributes": {
        "creationDate": "2023-11-07T22:13:06Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2023-11-07T23:59:01Z",
  "eventSource": "sqs.amazonaws.com",
  "eventName": "ChangeMessageVisibilityBatch",
  "awsRegion": "ap-southeast-4",
  "sourceIPAddress": "10.0.118.80",
  "userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
  "requestParameters": {
    "visibilityTimeout": 0,
    "entries": [
      {
        "id": "0",
        "receiptHandle":
 "AQEB2M5cVYg5gslhWME6537hdjcaPnOYPA5M0W460TTb0DzPle631yPWm8qxd4O1hDj/
B4ntTMnsgBTa95t14tNx7Vn96jKJ5rIoZ7iI8TRmkT1caKodKIPs8w9yndZq50c2FPQxtyH+2L3UHf/
abV3szqVWXOLZR4PwX8zZkWVQGNCNnY2q2lGCG586F8QwvrOFYoXNwB8ymd1t77e1PDPknq1Io3JFuzkEsndkkETy4fV
l5PHX17nXxaC+DURVlMPXOuSFACGmWqAoyk50HKwGOjLQgpySL/
TcnQXClvFq8kNXGwyVzJsbwHpOHxI7oce69vaD6DaWFP75d3hx+PJeG9pauQCKzVP3skt3Hw/
zDC7YfKcALD3aCwMmeNDwT3w0BUG6XZdG5lYhtFtTQYV7YuS3i/
Jh3HShGbtm07JKOEFiPkxv2+XNaAX3gFEpbng6zamTanfyMXCJIiglAEqiyWHQ=",
        "visibilityTimeout": 2271
      }
```

```
    ],
    "queueUrl": "https://sqs.ap-southeast-4.amazonaws.com/123456789012/MyQueue"
  },
  "responseElements": {
    "successful": [
      {
        "id": "0"
      }
    ]
  },
  "requestID": "d49ab65f-9dc7-54b8-875c-eb9b4c42988b",
  "eventID": "ca16c8c2-c4ba-4eb5-a54c-e650a10266d4",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::SQS::Queue",
      "ARN": "arn:aws:sqs:ap-southeast-4:123456789012:MyQueue"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": false,
  "recipientAccountId": "123456789012",
  "eventCategory": "Data",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "sqs.ap-southeast-4.amazonaws.com"
  }
}
```


Surveillance des files d'attente Amazon SQS à l'aide de CloudWatch

Amazon SQS et Amazon CloudWatch sont intégrés afin que vous puissiez les utiliser CloudWatch pour consulter et analyser les métriques de vos files d'attente Amazon SQS. Vous pouvez consulter et analyser les métriques de vos files d'attente depuis la console Amazon SQS, CloudWatch la console, en utilisant ou en utilisant AWS CLIl'API. CloudWatch Vous pouvez également définir des CloudWatch alarmes pour les métriques Amazon SQS.

CloudWatch les métriques relatives à vos files d'attente Amazon SQS sont automatiquement collectées et transmises à CloudWatch intervalles d'une minute. Ces statistiques sont collectées sur toutes les files d'attente qui répondent aux CloudWatch directives relatives à l'activité. CloudWatchconsidère qu'une file d'attente est active pendant six heures au maximum si elle contient des messages ou si une action y accède.

Lorsqu'une file d'attente Amazon SQS est inactive pendant plus de six heures, le service Amazon SQS est considéré comme inactif et cesse de fournir des métriques au service. CloudWatch Les données manquantes, ou les données représentant zéro, ne peuvent pas être visualisées dans les CloudWatch métriques d'Amazon SQS pendant la période pendant laquelle votre file d'attente Amazon SQS était inactive.

> **Note**
>
> - Une file d'attente Amazon SQS peut être activée lorsque l'utilisateur qui appelle une API depuis la file d'attente n'est pas autorisé et que la demande échoue.
> - La console Amazon SQS exécute un appel d'`GetQueueAttributes`API lorsque la page de la file d'attente est ouverte. La demande `GetQueueAttributes` d'API active la file d'attente.
> - Un délai pouvant aller jusqu'à 15 minutes se produit dans CloudWatch les métriques lorsqu'une file d'attente est activée à partir d'un état inactif.
> - Les métriques Amazon SQS rapportées dans le document sont gratuites. CloudWatch Elles sont fournies dans le cadre du service Amazon SQS.
> - CloudWatch les métriques sont prises en charge à la fois pour les files d'attente standard et FIFO.

Rubriques

Accès aux CloudWatch métriques pour Amazon SQS

Amazon SQS et Amazon CloudWatch sont intégrés afin que vous puissiez les utiliser CloudWatch pour consulter et analyser les métriques de vos files d'attente Amazon SQS. Vous pouvez consulter

et analyser les métriques de vos files d'attente depuis la console Amazon SQS, CloudWatch la console, en utilisant ou en utilisant AWS CLIl'API. CloudWatch Vous pouvez également définir des CloudWatch alarmes pour les métriques Amazon SQS.

Console Amazon SQS

1. Connectez-vous à la console Amazon SQS.
2. Dans la liste des files d'attente, sélectionnez (cochez) les files d'attente pour lesquelles vous souhaitez accéder aux métriques. Vous pouvez afficher les métriques de jusqu'à 10 files d'attente.
3. Sélectionnez l'onglet Monitoring (Surveillance).

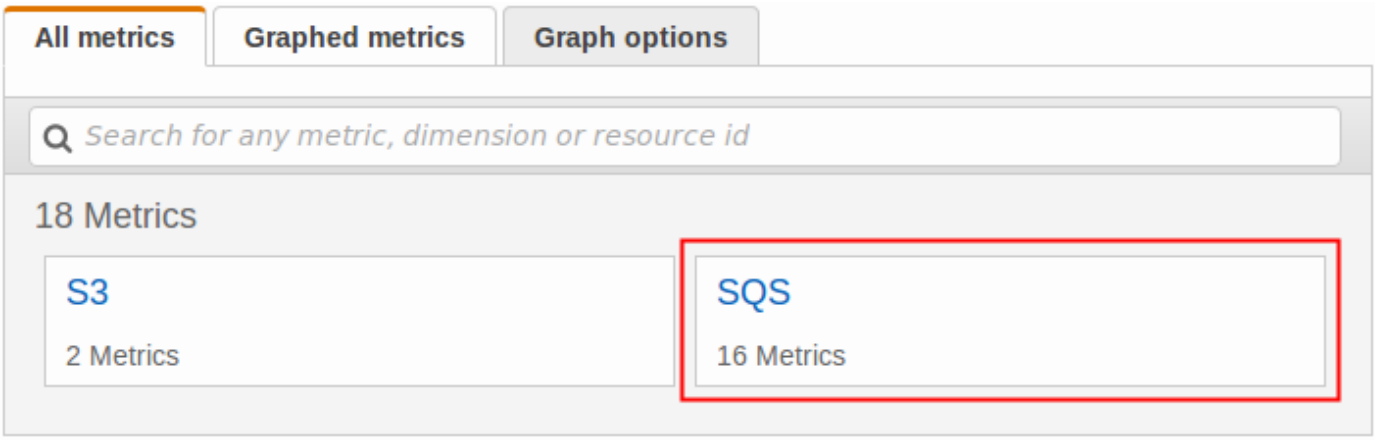
   Plusieurs graphiques sont affichés dans la section métriques SQS.
4. Pour comprendre un graphique particulier, passez la souris sur

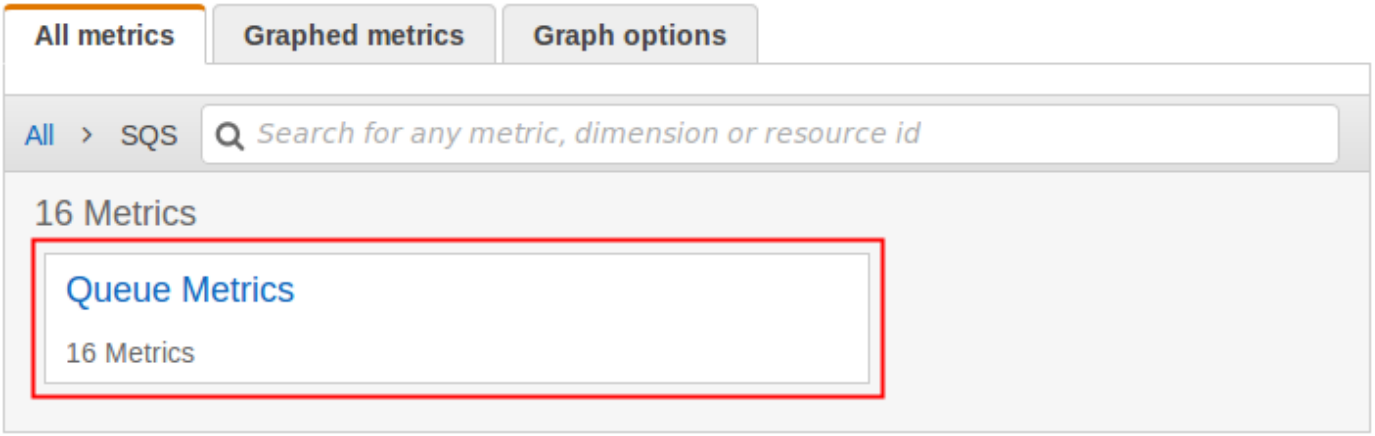
   en regard du graphique souhaité ou consultez CloudWatch Métriques disponibles pour Amazon SQS.
5. Pour modifier la plage de temps de tous les graphiques en même temps, sélectionnez la plage de temps souhaitée dans Plage de temps (par exemple, Dernière heure).
6. Pour afficher les statistiques supplémentaires d'un graphique individuel, sélectionnez ce dernier.
7. Dans la boîte de dialogue Détails de CloudWatch surveillance, sélectionnez une statistique (par exemple, Somme). Pour obtenir une liste des statistiques prises en charge, consultez la section CloudWatch Métriques disponibles pour Amazon SQS.
8. Pour modifier la plage de temps et l'intervalle de temps affichés par un graphique individuel (par exemple, pour afficher une plage de temps des 24 dernières heures au lieu des 5 dernières minutes, ou pour afficher une période de toutes les heures au lieu de toutes les 5 minutes), la boîte de dialogue du graphique étant toujours affichée, définissez la plage de temps souhaitée dans Plage de temps (par exemple, 24 dernières heures). Pour Période, sélectionnez la période souhaitée dans la plage de temps spécifiée (par exemple, 1 heure). Lorsque vous en avez terminé avec le graphe, cliquez sur Fermer.
9. (Facultatif) Pour utiliser des CloudWatch fonctionnalités supplémentaires, dans l'onglet Surveillance, choisissez Afficher toutes les CloudWatch mesures, puis suivez les instructions de la CloudWatch Console Amazon procédure.

CloudWatch Console Amazon

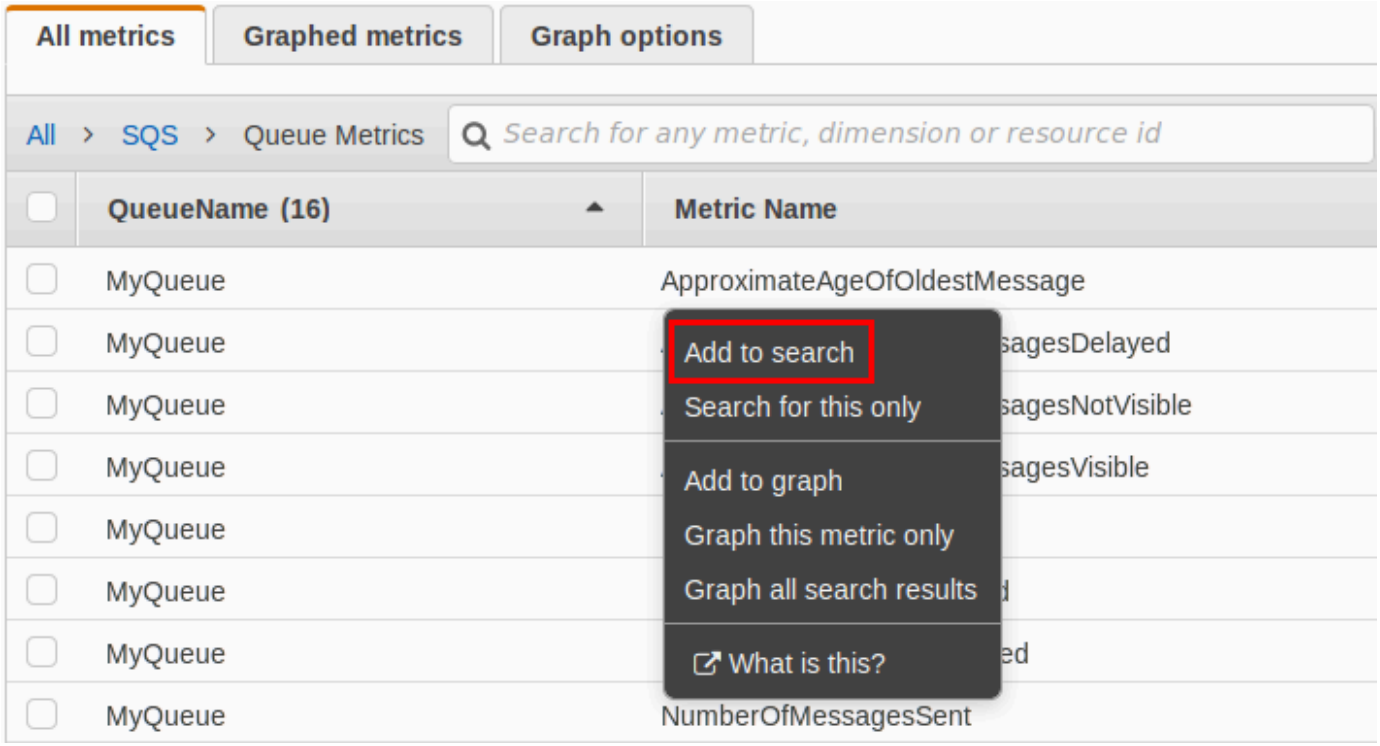
1. Connectez-vous à la console CloudWatch.
2. Dans le volet de navigation, choisissez Métriques.
3. Sélectionnez l'espace de nom de métrique SQS.



4. Sélectionnez la dimension de métrique Queue Metrics.



5. Vous pouvez désormais examiner vos métriques Amazon SQS :
   - Pour trier les métriques, utilisez l'en-tête de colonne.
   - Pour représenter graphiquement une métrique, cochez la case en regard de la métrique.
   - Pour filtrer par métrique, choisissez le nom de la métrique, puis Add to search (Ajouter à la recherche).



Pour plus d'informations et des options supplémentaires, consultez Graph Metrics et Using Amazon CloudWatch Dashboards dans le guide de l' CloudWatch utilisateur Amazon.

AWS Command Line Interface

Pour accéder aux métriques Amazon SQS à l'aide de AWS CLI, exécutez la `get-metric-statistics` commande.

Pour plus d'informations, consultez Obtenir des statistiques pour une métrique dans le guide de CloudWatch l'utilisateur Amazon.

CloudWatch API

Pour accéder aux métriques Amazon SQS à l'aide de l' CloudWatch API, utilisez l'`GetMetricStatistics`action.

Pour plus d'informations, consultez Obtenir des statistiques pour une métrique dans le guide de CloudWatch l'utilisateur Amazon.

Création d' CloudWatch alarmes pour les métriques Amazon SQS

CloudWatch vous permet de déclencher des alarmes en fonction d'un seuil métrique. Par exemple, vous pouvez créer une alarme pour la métrique `NumberOfMessagesSent`. Par exemple, si plus de 100 messages sont envoyés à la file d'attente `MyQueue` en 1 heure, une notification par e-mail est envoyée. Pour plus d'informations, consultez la section Création d' CloudWatch alarmes Amazon dans le guide de CloudWatch l'utilisateur Amazon.

1. Connectez-vous à la CloudWatch console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Choisissez Alarmes, puis Créer une alarme.
3. Dans la section Sélectionner une métrique de la boîte de dialogue Créer une alarme, choisissez Parcourir les métriques, SQS.
4. Pour SQS > Metrics de file d'attente, choisissez le QueueNamenom de la métrique pour laquelle vous souhaitez définir une alarme, puis choisissez Next. Pour obtenir la liste des métriques disponibles, consultez la section CloudWatch Métriques disponibles pour Amazon SQS.

   Dans l'exemple suivant, la sélection est destinée à une alarme pour la métrique `NumberOfMessagesSent` pour la file d'attente `MyQueue`. L'alarme se déclenche lorsque le nombre de messages envoyés dépasse 100.
5. Dans la section Définir une alarme de la boîte de dialogue Créer une alarme, procédez comme suit :

   a. Sous Seuil de l'alarme, tapez un Nom et une Description pour l'alarme.

   b. Définissez is sur > 100.

   c. Définissez pour sur 1 out of 1 datapoints (1 sur 1 point de données).

   d. Sous Aperçu de l'alarme, définissez Période sur 1 heure.

   e. Définissez Statistique sur Standard, Somme.

   f. Sous Actions, définissez Chaque fois que cette alarme, sur L'état est ALARME.

      Si vous souhaitez CloudWatch envoyer une notification lorsque l'alarme est déclenchée, sélectionnez une rubrique Amazon SNS existante ou choisissez Nouvelle liste et entrez les adresses e-mail séparées par des virgules.

> **Note**
>
> Si vous créez une rubrique Amazon SNS, les adresses e-mail doivent être vérifiées avant que vous ne puissiez recevoir des notifications. Si l'état de l'alarme change avant la vérification des adresses e-mail, les notifications ne sont pas remises.

6. Sélectionnez Create Alarm (Créer une alerte).

   L'alarme est créée.

CloudWatch Métriques disponibles pour Amazon SQS

Amazon SQS envoie les métriques suivantes à. CloudWatch

> **Note**
>
> Pour les files d'attente standard, le résultat est approximatif en raison de l'architecture distribuée d'Amazon SQS. Dans la plupart des cas, le nombre devrait être proche du nombre réel de messages dans la file d'attente.
> Pour les files d'attente FIFO, le résultat est exact.

Métriques Amazon SQS

L'espace de noms `AWS/SQS` inclut les métriques suivantes.

Métrique	Description
`ApproximateAgeOfOldestMessage`	Age approximatif du plus ancien message non supprimé dans la file d'attente. **Note** • Lorsqu'un message a été reçu trois fois (ou plus) et qu'il n'a pas été traité, il est déplacé

Métrique	Description
	à la fin de la file d'attente et la métrique `ApproximateAgeOfOldestMessage` pointe vers le deuxième message le plus ancien qui n'a pas été reçu plus de trois fois. Cette action se produit même si la file d'attente a une stratégie de redirection. • Un message « poison pill » (reçu plusieurs fois mais jamais supprimé) pouvant fausser cette métrique, l'âge d'un message de ce type n'est pas inclus dans la métrique tant que le message n'est pas consommé correctement. • Lorsque la file d'attente dispose d'une stratégie de redirection, le message est déplacé vers une file d'attente de lettres mortes après le nombre maximal de réceptions configuré. Lorsque le message est déplacé vers la file d'attente de lettres mortes, la métrique `ApproximateAgeOfOldestMessage` de la file d'attente de lettres mortes représente l'heure à laquelle le message a été déplacé vers la file d'attente de lettres mortes (et non l'heure

Métrique	Description
	d'origine à laquelle le message a été envoyé). • Pour les files d'attente FIFO, le message n'est pas déplacé à la fin de la file d'attente, car cela annulerait la garantie des commandes FIFO. Le message sera plutôt envoyé au DLQ s'il est configuré. Sinon, il bloquera le groupe de messages jusqu'à ce qu'il soit supprimé avec succès ou jusqu'à son expiratio n. Critères de rapport : une valeur non négative est signalée si la file d'attente est active. Unités : secondes Statistiques valides : Moyenne, Minimum, Maximum, Somme, Exemples de données (qui indique Exemple de comptage dans la console Amazon SQS)

Métrique	Description
`ApproximateNumberOfMessagesDelayed`	Nombre de messages dans la file d'attente qui sont retardés et qui ne peuvent pas être lus immédiatement. Cela peut se produire lorsque la file d'attente est configurée avec un délai d'attente ou que le message a été envoyé avec un paramètre de délai d'attente. Critères de rapport : une valeur non négative est signalée si la file d'attente est active. Unités : nombre Statistiques valides : Moyenne, Minimum, Maximum, Somme, Exemples de données (qui indique Exemple de comptage dans la console Amazon SQS)
`ApproximateNumberOfMessagesNotVisible`	Le nombre de messages « en vol ». Les messages sont considérés comme en cours s'ils ont été expédiés à un client, mais qu'ils n'ont pas encore été supprimés ou qu'ils n'ont pas encore atteint la fin du délai de visibilité. Critères de rapport : une valeur non négative est signalée si la file d'attente est active. Unités : nombre Statistiques valides : Moyenne, Minimum, Maximum, Somme, Exemples de données (qui indique Exemple de comptage dans la console Amazon SQS)

Métrique	Description
`ApproximateNumberOfMessagesVisible`	Nombre de messages à traiter. Critères de rapport : une valeur non négative est signalée si la file d'attente est active. Unités : nombre Statistiques valides : Moyenne, Minimum, Maximum, Somme, Exemples de données (qui indique Exemple de comptage dans la console Amazon SQS) Il n'y a aucune limite quant au nombre de messages à traiter, mais vous pouvez soumettre ce backlog à une période de conservation.
`NumberOfEmptyReceives` [1]	Nombre d'appels d'API `ReceiveMessage` qui n'ont pas renvoyé de message. Critères de rapport : une valeur non négative est signalée si la file d'attente est active. Unités : nombre Statistiques valides : Moyenne, Minimum, Maximum, Somme, Exemples de données (qui indique Exemple de comptage dans la console Amazon SQS)

Métrique	Description
`NumberOfMessagesDeleted` [1]	Nombre de messages supprimés de cette file d'attente. Critères de rapport : une valeur non négative est signalée si la file d'attente est active. Unités : nombre Statistiques valides : Moyenne, Minimum, Maximum, Somme, Exemples de données (qui indique Exemple de comptage dans la console Amazon SQS) Amazon SQS émet la métrique `NumberOfMessagesDeleted` pour chaque opération de suppression réussie qui utilise un descripteur de réception valide, y compris les suppressions en double. Dans les scénarios suivants, il est possible que la valeur de la métrique `NumberOfMessagesDeleted` soit plus élevée que prévu : • Appel de l'action `DeleteMessage` sur différents descripteurs de réception qui appartiennent au même message : si le message n'est pas traité avant l'expiration du délai de visibilité, le message devient disponible pour les autres consommateurs qui peuvent à nouveau le traiter et le supprimer, ce qui accroît la valeur de la métrique `NumberOfMessagesDeleted` . •

Métrique	Description
	Appel de l'action `DeleteMessage` sur le même descripteur de réception : si le message est traité et supprimé, mais que vous appelez à nouveau l'action `DeleteMessage` à l'aide du même descripteur de réception, l'état de réussite est renvoyé, ce qui accroît la valeur de la métrique `NumberOfMessagesDeleted` .
`NumberOfMessagesReceived` [1]	Nombre de messages renvoyés par les appels à l'action `ReceiveMessage` . Critères de rapport : une valeur non négative est signalée si la file d'attente est active. Unités : nombre Statistiques valides : Moyenne, Minimum, Maximum, Somme, Exemples de données (qui indique Exemple de comptage dans la console Amazon SQS)

Métrique	Description
`NumberOfMessagesSent` [1]	Nombre de messages ajoutés dans une file d'attente. Si vous envoyez un message à une file d'attente de lettres mortes manuellement, il est capturé par la métrique `NumberOfMessagesSent`. Toutefois, si un message est envoyé à une file d'attente de lettres mortes à la suite d'une tentative de traitement infructueuse, il n'est pas capturé par cette métrique. Par conséquent, il est possible que les valeurs des métriques `NumberOfMessagesSent` et `NumberOfMessagesReceived` diffèrent. Critères de rapport : une valeur non négative est signalée si la file d'attente est active. Unités : nombre Statistiques valides : Moyenne, Minimum, Maximum, Somme, Exemples de données (qui indique Exemple de comptage dans la console Amazon SQS)

Métrique	Description
`SentMessageSize` [1]	Taille des messages ajoutés à une file d'attente. Critères de rapport : une valeur non négative est signalée si la file d'attente est active. Unités : octets Statistiques valides : Moyenne, Minimum, Maximum, Somme, Exemples de données (qui indique Exemple de comptage dans la console Amazon SQS) **Note** `SentMessageSize` ne s'affiche pas en tant que métrique disponible dans la CloudWatch console tant qu'au moins un message n'est pas envoyé à la file d'attente correspondante.

[1] Ces métriques sont calculées du point de vue du service et peuvent inclure de nouvelles tentatives. Ne vous fiez pas aux valeurs absolues de ces métriques et ne les utilisez pas pour estimer l'état actuel de la file d'attente.

Dimensions pour les métriques Amazon SQS

La seule dimension à laquelle Amazon SQS envoie est. CloudWatch `QueueName` Cela signifie que toutes les statistiques disponibles sont filtrées par `QueueName`.

Validation de conformité pour Amazon SQS

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 Amazon SQS

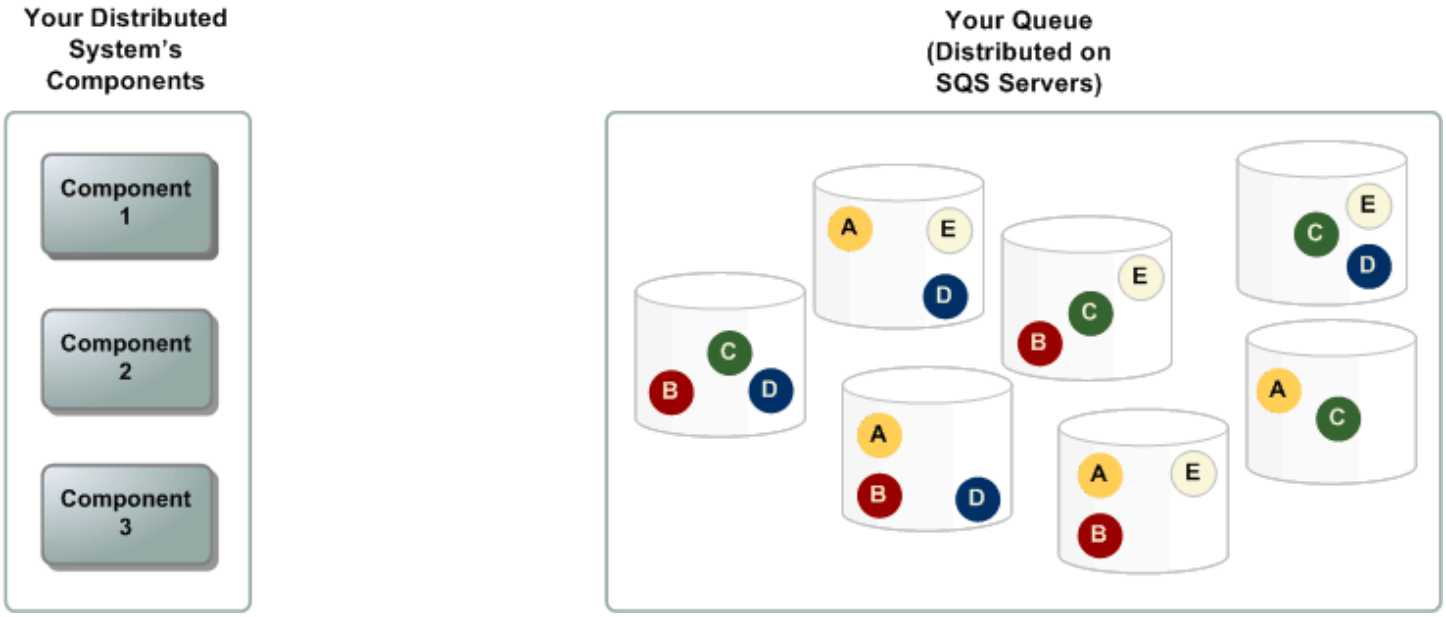
L'infrastructure AWS mondiale est construite autour des AWS régions et des zones de disponibilité. AWS Les régions fournissent plusieurs zones de disponibilité physiquement séparées et isolées, connectées par un réseau à faible latence, à haut débit et hautement redondant. Avec les zones de disponibilité, vous pouvez concevoir et exploiter des applications et des bases de données qui basculent automatiquement d'une zone à l'autre sans interruption. Les zones de disponibilité sont davantage disponibles, tolérantes aux pannes et ont une plus grande capacité de mise à l’échelle que les infrastructures traditionnelles à un ou plusieurs centres de données. Pour plus d'informations sur AWS les régions et les zones de disponibilité, consultez la section Infrastructure AWS mondiale.

Outre l'infrastructure AWS mondiale, Amazon SQS propose des files d'attente distribuées.

Files d'attente distribuées

Un système de messagerie distribué comprend trois éléments principaux : les composants de votre système distribué, votre file d'attente (distribuée sur des serveurs Amazon SQS) et les messages de la file d'attente.

Dans le scénario suivant, le système comprend plusieurs producteurs (composants qui envoient des messages à la file d'attente) et plusieurs consommateurs (composants qui reçoivent des messages de la file d'attente). La file d'attente (qui contient les messages A à E) stocke les messages de manière redondante sur plusieurs serveurs Amazon SQS.



Sécurité de l'infrastructure dans Amazon SQS

En tant que service géré, Amazon SQS est protégé par les procédures de sécurité du réseau AWS mondial décrites dans le livre blanc Amazon Web Services : présentation des processus de sécurité.

Vous utilisez les actions d'API AWS publiées pour accéder à Amazon SQS via le réseau. Les clients doivent prendre en charge le protocole TLS (Transport Layer Security) 1.2 ou version ultérieure. Les clients doivent également prendre en charge les suites de chiffrement PFS (Perfect Forward Secrecy) comme Ephemeral Diffie-Hellman (DHE) ou Elliptic Curve Ephemeral Diffie-Hellman (ECDHE)

En outre, vous devez signer les demandes à l'aide d'un ID de clé d'accès et d'une clé d'accès secrète associée à un mandataire IAM. Vous pouvez également utiliser AWS Security Token Service (AWS STS) pour générer des informations d'identification de sécurité temporaires afin de signer les demandes.

Vous pouvez appeler ces actions d'API à partir de n'importe quel emplacement sur le réseau, mais Amazon SQS prend en charge les stratégies d'accès basées sur les ressources, ce qui peut inclure des restrictions en fonction de l'adresse IP source. Vous pouvez également utiliser des stratégies Amazon SQS pour contrôler l'accès à partir de points de terminaison d'un VPC Amazon ou de VPC spécifiques. Cela permet d'isoler efficacement l'accès réseau à une file d'attente Amazon SQS donnée uniquement du VPC spécifique au sein du réseau. AWS Pour plus d'informations, voir Exemple 5 : Refuser l'accès s'il n'émane pas d'un point de terminaison de VPC.

Bonnes pratiques de sécurité pour Amazon SQS

AWS fournit de nombreuses fonctionnalités de sécurité pour Amazon SQS, que vous devriez examiner dans le contexte de votre propre politique de sécurité. Voici les bonnes pratiques de sécurité préventive pour Amazon SQS.

> **Note**
>
> Les conseil de mise en œuvre spécifiques fournies concernent les implémentations et les cas d'utilisation courants. Nous vous suggérons de consulter ces bonnes pratiques dans le contexte de votre cas d'utilisation, de votre architecture et de votre modèle de menace spécifique.

Rubriques

S'assurer que les files d'attente ne sont pas accessibles publiquement

À moins que vous ne demandiez explicitement à quiconque sur Internet de lire ou d'écrire dans votre file d'attente Amazon SQS, vous devez vous assurer que votre file d'attente n'est pas accessible au public (accessible par tout le monde dans le monde ou par tout utilisateur authentifié AWS).

- Évitez de créer des stratégies avec `Principal` défini sur `""`.
- Évitez d'utiliser un caractère générique (`*`). Nommez plutôt un ou plusieurs utilisateurs spécifiques.

Implémentation d'un accès sur la base du moindre privilège

Lorsque vous accordez des autorisations, vous décidez qui les reçoit, pour quelles files d'attente celles-ci sont destinées et quelles actions d'API spécifiques vous souhaitez autoriser pour ces files

d'attente. La mise en œuvre du moindre privilège est importante pour réduire les risques de sécurité et atténuer l'effet des erreurs ou des intentions malveillantes.

Suivez les conseils de sécurité standard pour accorder le moindre privilège. Autrement dit, accordez uniquement les autorisations requises pour effectuer une tâche spécifique. Vous pouvez implémenter ceci à l'aide d'une combinaison de stratégies de sécurité.

Amazon SQS utilise le modèle producteur-consommateur nécessitant trois types d'accès de compte utilisateur :

- Administrateurs - Accès à la création, à la modification et à la suppression de files d'attente. Les administrateurs contrôlent également les stratégies de file d'attente.
- Producteurs : accès à l'envoi de messages dans les files d'attente.
- Consommateurs : accès à la réception et à la suppression des messages dans les files d'attente.

Pour plus d'informations, consultez les sections suivantes :

- Gestion des identités et des accès dans Amazon SQS
- Autorisations d'API Amazon SQS : référence des actions et ressources
- Utilisation de stratégies personnalisées avec le langage de la stratégie d'accès Amazon SQS

Utiliser les rôles IAM pour les applications et les AWS services qui nécessitent un accès Amazon SQS

Pour que des applications ou AWS des services tels qu'Amazon EC2 puissent accéder aux files d'attente Amazon SQS, ils doivent utiliser des informations d'identification AWS valides dans leurs demandes d'API. AWS Ces informations d'identification ne faisant pas l'objet d'une rotation automatique, vous ne devez pas les stocker AWS directement dans l'application ou l'instance EC2.

Vous devez utiliser un rôle IAM pour gérer des informations d'identification temporaires pour les applications ou services devant accéder à Amazon SQS. Lorsque vous utilisez un rôle, vous n'avez pas à distribuer des informations d'identification à long terme (telles qu'un nom d'utilisateur, un mot de passe et des clés d'accès) à une instance ou à un AWS service EC2 tel que AWS Lambda. Le rôle fournit plutôt des autorisations temporaires que les applications peuvent utiliser lorsqu'elles appellent d'autres AWS ressources.

Pour de plus amples informations, veuillez consulter Rôles IAMet Scénarios courants pour les rôles : utilisateurs, applications et services dans le guide de l'utilisateur.

Mise en œuvre du chiffrement côté serveur

Pour atténuer les problèmes de fuite de données, utilisez le chiffrement au repos pour chiffrer vos messages à l'aide d'une clé stockée dans un emplacement différent de celui où les messages sont stockés. Le chiffrement côté serveur (SSE) fournit le chiffrement des données au repos. Amazon SQS chiffre vos données au niveau des messages lorsqu'il les stocke et déchiffre les messages pour vous lorsque vous y accédez. SSE utilise des clés gérées dans AWS Key Management Service. Tant que vous authentifiez votre demande et que vous avez des autorisations d'accès, il n'y a aucune différence entre l'accès aux données chiffrées et aux données déchiffrées.

Pour plus d'informations, consultez Chiffrement au repos dans Amazon SQS et Gestion des clés Amazon SQS.

Application du chiffrement des données en transit

Sans HTTPS (TLS), un attaquant basé sur le réseau peut espionner le trafic réseau ou le manipuler, en utilisant une attaque telle que. man-in-the-middle Autorisez uniquement les connexions chiffrées via HTTPS (TLS) en utilisant la condition `aws:SecureTransport` de la stratégie de file d'attente pour forcer les demandes à utiliser SSL.

Réflexion sur l'utilisation des points de terminaison de VPC pour accéder à Amazon SQS

Si vous avez des files d'attente avec lesquelles vous devez pouvoir interagir mais qui ne doivent absolument pas être exposées à Internet, utilisez des points de terminaison de VPC pour mettre en file d'attente l'accès uniquement aux hôtes au sein d'un VPC particulier. Vous pouvez utiliser des stratégies de file d'attente pour contrôler l'accès aux files d'attente à partir de points de terminaison d'un VPC Amazon spécifiques ou de VPC spécifiques.

Les points de terminaison d'un VPC Amazon SQS offrent deux façons de contrôler l'accès à vos messages :

- Vous pouvez contrôler les demandes, les utilisateurs ou les groupes autorisés à traverser un point de terminaison d'un VPC spécifique.
- Vous pouvez contrôler quels VPC ou points de terminaison de VPC ont accès à votre file d'attente à l'aide d'une stratégie de file d'attente.

Pour plus d'informations, consultez Points de terminaison Amazon Virtual Private Cloud pour Amazon SQS et Création d'une stratégie de point de terminaison d'un VPC Amazon pour Amazon SQS.

Ressources Amazon SQS associées

Le tableau suivant répertorie les ressources connexes qui peuvent vous être utiles lors de l'utilisation de ce service.

Ressource	Description
Référence d'API Amazon Simple Queue Service	Descriptions des actions, paramètres et types de données, et liste des erreurs renvoyées par le service.
Amazon SQS dans la référence des commandes de l'AWS CLI	Descriptions des AWS CLI commandes que vous pouvez utiliser pour gérer les files d'attente.
Régions et points de terminaison	Informations sur les régions et les points de terminaison Amazon SQS
Page produit	Page web principale pour de plus amples informations sur Amazon SQS.
Forum de discussion	Un forum communautaire pour les développeurs où ils peuvent discuter des questions techniques liées à Amazon SQS.
AWS Informations sur le Support Premium	Page Web principale contenant des informations sur le Support AWS Premium, un canal d'assistance personnalisé et rapide destiné à vous aider à créer et à exécuter des applications sur AWS des services d'infrastructure.

Historique de la documentation

Le tableau suivant décrit les modifications importantes apportées au Guide du développeur Amazon Simple Queue Service depuis janvier 2019. Pour recevoir les notifications des mises à jour de cette documentation, abonnez-vous au flux RSS.

Les fonctionnalités du service sont parfois déployées progressivement AWS dans les régions où un service est disponible. Nous mettons à jour cette documentation pour la première version uniquement. Nous ne fournissons pas d'informations sur la disponibilité des régions et n'annonçons pas les déploiements régionaux ultérieurs. Pour plus d'informations sur la disponibilité des fonctionnalités du service dans les régions et pour vous abonner aux notifications concernant les mises à jour, voir Quelles sont les nouveautés AWS ? .

Modification	Description	Date
AWS Protocole JSON	Effectuez des demandes d'API à l'aide du protocole AWS JSON.	27 juillet 2023
Nouvelle section décrivant les politiques AWS gérées pour Amazon SQS et les mises à jour de ces politiques	Amazon SQS a ajouté une nouvelle action qui vous permet de répertorier les tâches de transfert de messages les plus récentes (jusqu'à 10) dans une file d'attente source spécifique. Cette action est associée à l'opération d'API `ListMessageMoveTasks` .	7 juin 2023
Redirection des files d'attente de lettres mortes à l'aide d'API	Configurez les redirections des files d'attente de lettres mortes à l'aide des API Amazon SQS.	7 juin 2023
ABAC pour Amazon SQS	Contrôle d'accès par attributs (ABAC) à l'aide de balises de file d'attente pour des	10 novembre 2022

	autorisations d'accès flexibles et évolutives.	
La limite de débit élevé FIFO augmente	Augmentation des quotas par défaut pour le mode débit élevé FIFO dans les régions commerciales, ainsi que pour l'optimisation des documents à haut débit FIFO.	20 octobre 2022
Le chiffrement côté serveur (SSE) par défaut est disponible	Chiffrement côté serveur (SSE) à l'aide du chiffrement propre à SQS (SSE-SQS) par défaut.	26 septembre 2022
La prise en charge de la protection contre le problème de député confus d'Amazon SQS est disponible	La protection contre le problème de député confus vous permet de spécifier de nouveaux en-têtes dans les demandes, qui sont vérifiés par rapport aux conditions de la stratégie KMS lors de l'utilisation du SSE géré par Amazon SQS.	29 décembre 2021
Le chiffrement SSE géré est disponible	Le chiffrement SSE géré par Amazon SQS (SSE-SQS) est un chiffrement géré côté serveur qui utilise des clés de chiffrement appartenant à Amazon SQS pour protéger les données sensibles envoyées via des files d'attente de messages.	23 novembre 2021

La redirection des files d'attente de lettres mortes est disponible	Amazon SQS prend en charge la redirection des files d'attente de lettres mortes pour les files d'attente standard.	10 novembre 2021
Un débit élevé pour les messages dans les files d'attente FIFO est disponible	Le débit élevé pour les files d'attente FIFO Amazon SQS permet d'augmenter le nombre de transactions par seconde (TPS) pour les messages dans les files d'attente FIFO. Pour plus d'informations sur les quotas de débit, consultez la section Quotas relatifs aux messages.	27 mai 2021
Un débit élevé pour les messages dans les files d'attente FIFO est disponible dans la version préliminaire	Le débit élevé pour les files d'attente FIFO Amazon SQS est en version préliminaire et est susceptible d'être modifié. Cette fonctionnalité fournit un plus grand nombre de transactions par seconde (TPS) pour les messages dans les files d'attente FIFO. Pour plus d'informations sur les quotas de débit, consultez la section Quotas relatifs aux messages.	17 décembre 2020
Nouvelle conception de la console Amazon SQS	Pour simplifier les flux de travail de développement et de production, la console Amazon SQS propose une nouvelle expérience utilisateur.	8 juillet 2020

Amazon SQS prend en charge la pagination pour les ListQueues et listDeadLetter SourceQueues	Vous pouvez spécifier le nombre maximum de résultats à renvoyer à partir d'une ListQueues ou d'une demande de liste. DeadLetter SourceQueues	22 juin 2020
Amazon SQS prend en charge les métriques CloudWatch Amazon d'une minute dans AWS toutes les régions, à AWS GovCloud l'exception des régions (États-Unis)	La CloudWatch métrique d'une minute pour Amazon SQS est disponible dans toutes les régions, à l'exception AWS GovCloud (US) des régions.	9 janvier 2020
Amazon SQS prend en charge les métriques d'une minute CloudWatch	La CloudWatch métrique d'une minute pour Amazon SQS n'est actuellement disponibl e que dans les régions suivantes : USA Est (Ohio), Europe (Irlande), Europe (Stockholm) et Asie-Pacifique (Tokyo).	25 novembre 2019
AWS Lambda des déclenche urs pour les files d'attente FIFO Amazon SQS sont disponibles	Vous pouvez configurer les messages arrivant dans une file d'attente FIFO en tant que déclencheur d'une fonction Lambda.	25 novembre 2019
Le chiffrement côté serveur (SSE) pour Amazon SQS est disponible dans les régions de Chine	SSE pour Amazon SQS est disponible dans les régions de Chine.	13 novembre 2019
Les files d'attente FIFO sont disponibles dans la région Moyen-Orient (Bahreïn)	Les files d'attente FIFO sont disponibles dans la région Moyen-Orient (Bahreïn).	10 octobre 2019

Les points de terminaison Amazon Virtual Private Cloud (Amazon VPC) pour Amazon SQS sont disponibles dans les régions AWS GovCloud (USA Est) et (USA Ouest) AWS GovCloud	Vous pouvez envoyer des messages à vos files d'attente Amazon SQS depuis Amazon VPC dans les régions AWS GovCloud (USA Est) et (USA Ouest). AWS GovCloud	5 septembre 2019
Amazon SQS permet de résoudre les problèmes de files d'attente à l' AWS X-Ray aide des attributs du système de messagerie	Vous pouvez résoudre les problèmes liés aux messages transitant par des files d'attente Amazon SQS à l'aide de X-Ray. Cette version ajoute le paramètre de demande `MessageSystemAttribute` (qui vous permet d'envoyer des en-têtes de suivi X-Ray via Amazon SQS) aux opérations d'API `SendMessage` et `SendMessageBatch`, l'attribut `AWSTraceHeader` à l'opération d'API `ReceiveMessage` et le type de données `MessageSystemAttributeValue`.	28 août 2019

Vous pouvez baliser les files d'attente Amazon SQS lors de leur création	Vous pouvez utiliser un seul appel d'API Amazon SQS, une fonction AWS SDK ou une commande AWS Command Line Interface (AWS CLI) pour créer simultanément une file d'attente et spécifier ses balises. En outre, Amazon SQS prend en charge les clés `aws:TagKeys` et `aws:RequestTag` AWS Identity and Access Management (IAM).	22 août 2019
Le client de file d'attente temporaire pour Amazon SQS est désormais disponible	Les files d'attente temporair es vous permettent de gagner du temps de développement et de réduire les coûts de déploiement lorsque vous utilisez des modèles de messages courants tels que demande-réponse. Vous pouvez utiliser le Client de file d'attente temporaire pour créer des files d'attente temporaires à haut débit, économiques et gérées par l'application.	25 juillet 2019
SSE pour Amazon SQS est disponible dans la région AWS GovCloud (USA Est)	Le chiffrement côté serveur (SSE) pour Amazon SQS est disponible dans AWS GovCloud la région (USA Est).	20 juin 2019

Les files d'attente FIFO sont disponibles dans les régions Asie-Pacifique (Hong Kong), Chine (Pékin), AWS GovCloud (USA Est) et AWS GovCloud (USA Ouest)	Les files d'attente FIFO sont disponibles dans les régions Asie-Pacifique (Hong Kong), Chine (Pékin), AWS GovCloud (USA Est) et AWS GovCloud (USA Ouest).	15 mai 2019
Les stratégies de point de terminaison d'un VPC Amazon sont disponibles pour Amazon SQS	Vous pouvez créer des stratégies de point de terminaison d'un VPC Amazon pour Amazon SQS.	4 avril 2019
Les files d'attente FIFO sont disponibles dans les régions Europe (Stockholm) et Chine (Ningxia)	Les files d'attente FIFO sont disponibles dans les régions Europe (Stockholm) et Chine (Ningxia).	14 mars 2019
Les files d'attente FIFO sont disponibles dans toutes les régions où Amazon SQS est disponible	Les files d'attente FIFO sont disponibles dans les régions USA Est (Ohio), USA Est (Virginie du Nord), USA Ouest (Californie du Nord), USA Ouest (Oregon), Asie-Pacifique (Mumbai), Asie-Pacifique (Séoul), Asie-Pacifique (Singapour), Asie-Pacifique (Sydney), Asie-Pacifique (Tokyo), Canada (Centre), Europe (Francfort), Europe (Irlande), Europe (Londres), Europe (Paris) et Amérique du Sud (São Paulo).	7 février 2019

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.